

Bản dịch Datasheet của vi điều khiển AVR Atmega 128

Created by Le Duy Khanh

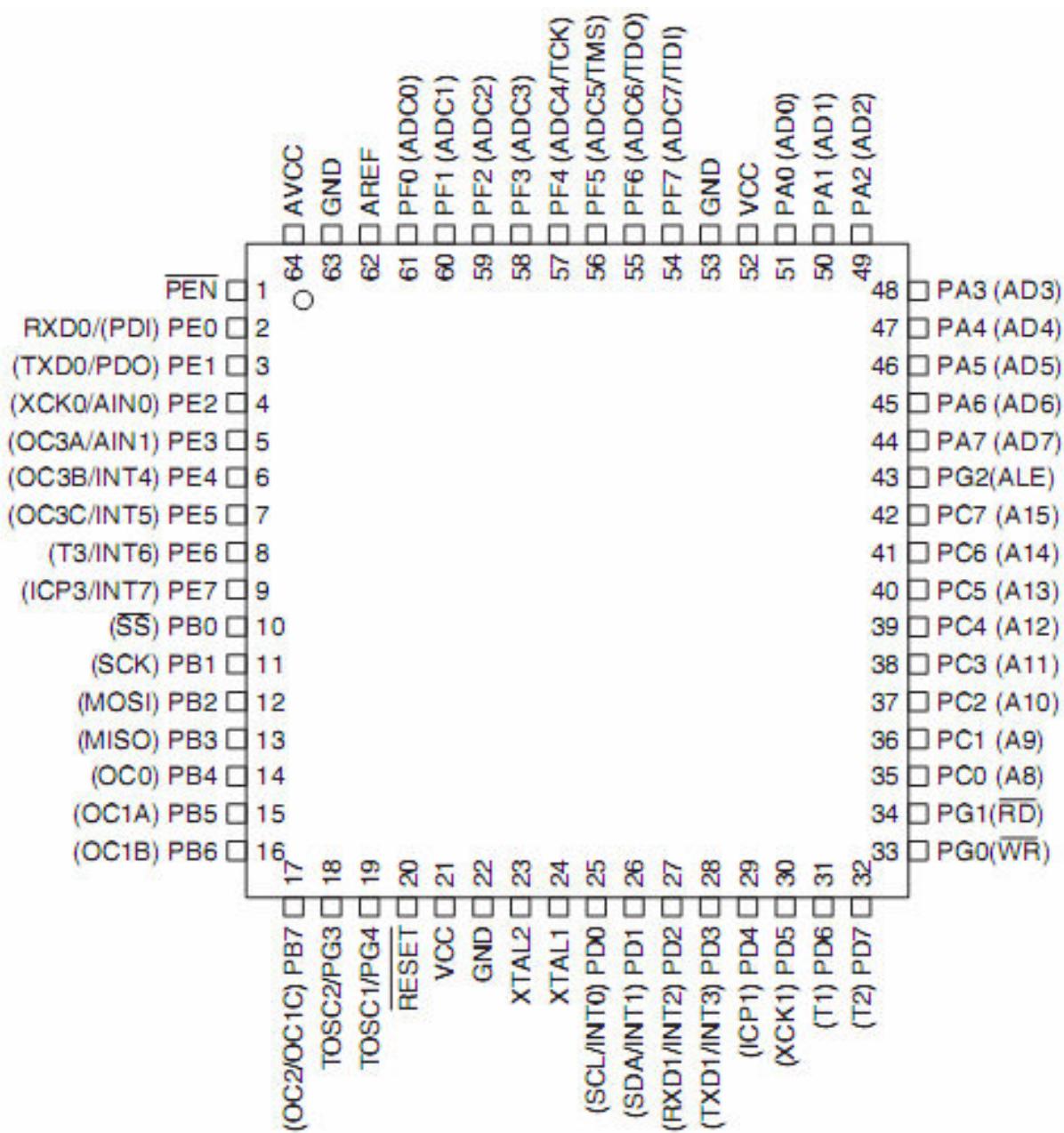
Date : 16/02/2010

## I . Đặc điểm , tính năng ( Features ) :

- + Hiệu suất cao , tiết kiệm điện
- + Hoàn thiện cấu trúc RISC
  - 133 lệnh hiệu quả - thực thi tất cả các chu kỳ đồng hồ đơn
  - 32 \*8 thanh ghi chung đa năng + các thanh ghi điều khiển ngoại vi
  - Đầy đủ các quá trình điều khiển tĩnh
  - Nâng lên 16 MIPS dữ liệu tại 16 MHz
  - Chip 2 nhân
- + Độ bền , sức chịu đựng cao , không thay đổi phân vùng nhớ
  - 128 K Bytes bộ nhớ Flash có thể lập trình được trong hệ thống
  - 4K Bytes EEPROM
  - 4K Bytes bộ nhớ SRAM bên trong
  - Chu kỳ ghi/xóa : 10000 Flash / 100000 EEPROM
  - Độ bền dữ liệu 20 năm ở 85 độ / 100 năm ở 25 độ
  - Đoạn mã lựa chọn chế độ khởi động với các bít khóa đọc lập trong chương trình hệ thống bởi chương trình khởi động đọc thật trong khi quá trình ghi diễn ra
  - Tối đa 64K Bytes không gian nhớ bên ngoài lựa chọn
  - Lập trình khóa cho phần mềm bảo mật
  - Giao diện SPI cho lập trình trong hệ thống
- + giao diện JTAG ( phù hợp với tiêu chuẩn IEEE 1149.1)
  - Khả năng quét biên theo tiêu chuẩn JTAG
  - Hỗ trợ chế độ sửa tạm ( debug ) trên chip
  - Lập trình của Flash , EEPROM , bộ bảo vệ ( FUSE) và Bit khóa ( Lock Bits) thông qua giao diện JTAG
- + Đặc điểm ngoại vi
  - 2 bộ Timer /counter 8 bit với bộ đếm gộp trước riêng biệt và chế độ so sánh mẫu
  - 2 bộ timer /counter 16 bit mở rộng với bộ đếm gộp trước chế độ so sánh mẫu và chế độ thu thập ( bắt dữ liệu )
  - Bộ counter thời gian thực với bộ dao động ( oscillator ) riêng biệt
  - 2 kênh PWM 8 bit
  - 6 kênh PWM với khả năng lập trình chính xác từ 2 đến 16 bit
  - Bộ điều chế so sánh tín hiệu ra

- 8 kênh , 10 bit ADC : 8 kênh đầu cuối đơn , 7 kênh khác nhau ( vi phân ) , 2 kênh khác nhau với bộ khuyêch đại lập trình được tại 1x , 10x ,200x
  - Bit định hướng với 2 dây giao diện nối tiếp
  - Lập trình kép các USARTs nối tiếp
  - Giao diện nối tiếp SPI chủ tớ
  - Lập trình timer Watchdog với bộ dao động trên chip
  - Bộ so sánh tương tự trên chip
- + các tính năng đặc biệt của bộ vi xử lí
- thiết lập bật lại nguồn và lập trình lại khi phát hiện nguồn yếu (brown-out)
  - hiệu chỉnh bộ dao động RC bên trong
  - Ngắt nguồn trong và ngoài
  - 6 chế độ chờ ( sleep ) : Idle nghỉ , giảm ồn ADC , tiết kiệm điện ( power – saver) , ngắt điện , chế độ chờ ( standby ) , chế độ chờ mở rộng
  - Phần mềm lựa chọn tần số xung nhịp
  - Lựa chọn chế độ so sánh Atmega 103 bởi bộ cầu chì Fuse
  - Vô hiệu hóa dừng lại toàn bộ
- + cổng vào ra và dạng đóng gói
- 53 đường vào ra lập trình được
  - 64 chân TQFP và 64 khói QFN/MLF
- + Điện áp hoạt động
- 2,7 – 5,5 V Atmega 128L
  - 4,5 – 5,5 V Atmega 128
- + Mức tốc độ xung nhịp
- 0 – 8 MHz Atmega 128L
  - 0 – 16 Mhz Atmega 128

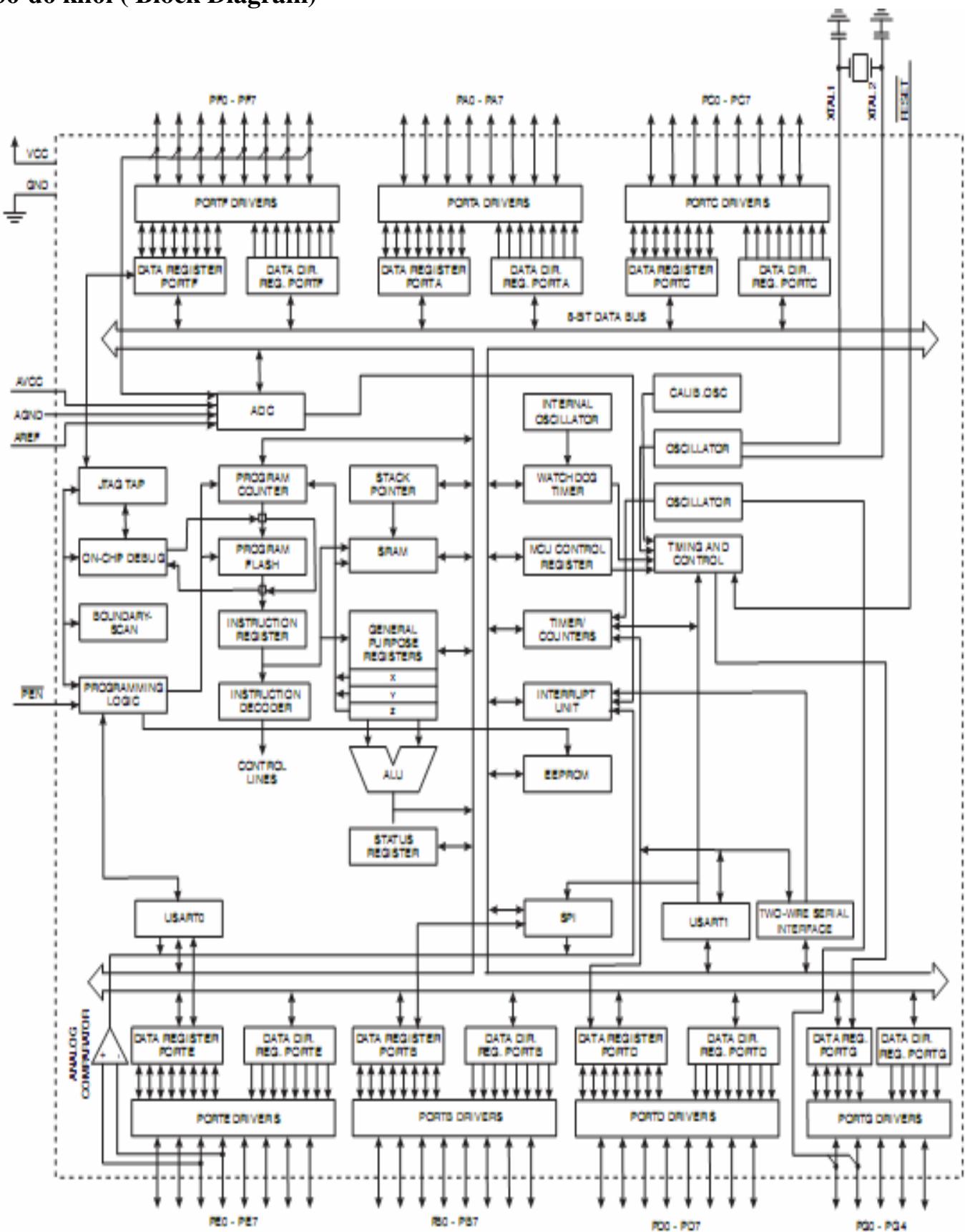
## II . Cấu hình chân ( pin configurations )



Hình 1 : chân ra của Atmega 128

Atmega 128 là một bộ vi xử lý CMOS điện áp thấp dựa trên nền kiến trúc AVR RISC nâng cao. Bằng cách thi hành các lệnh một cách mạnh mẽ trong một chu kỳ đồng hồ duy nhất, Atmega128 có thể cho phép tốc độ đạt được là 1 MIPS trên 1 MHz từ đó nó giúp người thiết kế hệ thống có khả năng tối ưu hóa điện năng sử dụng so với tốc độ xử lý.

## Sơ đồ khối ( Block Diagram)



Lõi AVR bao gồm 1 tập hợp các lệnh cài đặt với 32 thanh ghi chung đa năng . Tất cả 32 thanh ghi thì được nối trực tiếp với khối số học và logic (ALU ) nó cho phép 2 thanh ghi độc lập được truy cập trong 1 lệnh thực thi trong một chu kì quét xung đồng hồ . kết quả của cấu trúc này là có nhiều kiểu chế độ hiệu quả hơn trong khi vẫn đạt được tốc độ tối đa nhanh hơn 10 lần các bộ vi xử lý CISC thông thường .

Atmega 128 cũng cung cấp các tính năng sau đây : 128K bytes của bộ nhớ lập trình Flash trong hệ thống với khả năng đọc trong khi đang ghi , 4 K bytes EEPROM , 4 K bytes SRAM , 53 đường vào ra đa năng , 32 thanh ghi chung đa năng , bộ đếm thời gian thực , 4 bộ timer /counter tiện dụng với kiểu so sánh và PWM , 2 USART , 1 bit định hướng 2 dây giao diện nối tiếp , 8 kênh , 10 bit ADC với các lựa chọn các cổng vào riêng biệt với khả năng lập trình khuyêch đại , lập trình timer Watchdog với bộ tạo dao động bên trong , 1 cổng SPI nối tiếp , phù hợp với chuẩn IEEE 1149.1 , giao diện kiểm tra JTAG , cũng sử dụng để truy cập vào chế độ sửa tạm hệ thống trên chip và hỗ trợ lập trình , và 6 phần mềm có thể lựa chọn chế độ tiết kiệm điện . Chế độ rỗi IDLE dừng CPU trong khi cho phép SRAM , Timer/counter , cổng SPI , và các ngắt hệ thống tiếp tục vận hành . Chế độ tắt nguồn tiết kiệm dung lượng của thanh ghi nhưng nó làm đóng băng bộ tạo dao động (oscillator ) bên trong , vô hiệu hóa tất cả các chức năng của chip cho đến khi có ngắt kế tiếp hoặc là reset lại phần cứng (reset hardware ) . Trong chế độ tiết kiệm điện timer dị bộ vẫn tiếp tục chạy , điều này cho phép người sử dụng bảo dưỡng trong một thời gian trong khi phần còn lại của thiết bị đang trong quá trình nghỉ ( ngủ ) sleeping. Kiểu giảm nhiễu ADC dừng CPU và tắt cả các modul vào ra trừ các timer dị bộ và ADC , làm cực tiểu nhiễu chuyển mạch trong suốt quá trình chuyển đổi ADC . Trong chế độ chờ Standby bộ tạo dao động thạch anh và cộng hưởng đang chạy trong khi phần còn lại của thiết bị đang trong trạng thái ngủ . Điều này cho phép rất nhiều các khởi tạo nhanh được đồng thời tiêu thụ điện thấp . Trong chế độ chờ mở rộng , cả hai bộ tạo dao động chính và các timer dị bộ vẫn tiếp tục chạy

Thiết bị này được sản xuất dựa trên công nghệ chíp nhớ độ đặc cao của ATMEL . Bộ nhớ flash ISP trên chip cho phép bộ nhớ chương trình được lập trình lại trong hệ thống thông qua 1 giao diện ISP nối tiếp , bằng một chương trình lập trình bộ nhớ cố định thông thường hoặc bằng một chương trình khởi động đang chạy trong lõi của AVR . Chương trình khởi động có thể sử dụng bất cứ giao diện nào để tải các chương trình ứng dụng vào trong bộ nhớ các chương trình ứng dụng ( Flash ) . Phần mềm trong phần vùng khởi động của bộ nhớ Flash sẽ tiếp tục chạy trong khi các ứng dụng trong phần vùng này được cập nhật , cung cấp hoạt động đọc trong khi ghi . Bằng việc kết hợp 1 CPU- cấu trúc 8 bit RISC với bộ nhớ flash lập trình hệ thống trên 1 chip đơn , Atmega 128 là một vi xử lý mạnh , nó cung cấp 1 sự linh hoạt cao và môi trường làm việc có ích cho rất nhiều các ứng dụng điều khiển nhúng .

Atmega 128 được hỗ trợ với 1 sự thích hợp đầy đủ của chương trình và các công cụ phát triển hệ thống bao gồm : trình biên dịch C , các macro Asemmbler , các chương

trình chạy thử và mô phỏng , 1 bộ mô phỏng mạch điện , và các công cụ đánh giá so sánh

## Sự tương thích với Atmega 103 và Atmega 128

Atmega 128 là một vi xử lý có độ phức tạp cao mà ở đó số đầu vào ra được tích hợp rất nhiều lên đến 64 địa chỉ vào ra được dự trữ sẵn trong các lệnh cài đặt . Để đảm bảo tương thích với Atmega 103 thì tất cả các địa chỉ vào ra hiện nay trong Atmega 103 đều giống địa chỉ của Atmega 128 . Tất cả các địa chỉ I/O thêm vào thì được thêm vào trong một không gian địa chỉ I/O mở rộng bắt đầu từ \$60 đến \$FF ( trong Atmega 103 thì chưa trong không gian của RAM trong ) . Những địa chỉ này có thể được gọi bằng việc chỉ sử dụng các lệnh LD/LDS/LDD và ST/STS/STD , không phải sử dụng các lệnh IN hoặc OUT . Việc đặt lại các địa chỉ trong RAM của Atmega 103 có thể vẫn là một vấn đề cho người sử dụng . Ngoài ra , sự gia tăng về số lượng các vecto ngắt có thể là một vấn đề nếu các mã sử dụng là địa chỉ tuyệt đối . Để giải quyết vấn đề này , một sự tương thích của Atmega 103 có thể được lựa chọn bởi việc lập trình cho Fuse M103C . Trong chế độ này , không có chức năng nào ở trong không gian I/O mở rộng được sử dụng , vì vậy RAM trong được đặt địa chỉ như của Atmega 103 . Ngoài ra , các vecto ngắt mở rộng được gỡ bỏ .

Atmega 128 thì thích hợp 100 % với Atmega 103 , và có thể thay thế cho Atmega 103 trên cùng một bo mạch in hiện hành . chú ý ứng dụng " sự thay thế Atmega 103 bằng Atmega 128 " mô tả cái mà người sử dụng nên nhận biết sự thay thế của Atmega 103 bằng Atmega 128 .

## Chế độ tương thích của Atmega 103

Bằng việc lập trình M130C , Atmega 128 sẽ tương thích với Atmega 103 để ý ở RAM , chân I/O và các vecto ngắt được miêu tả như ở dưới đây . Tuy nhiên , một vài đặc điểm ở Atmega 128 thì không có ích trong chế độ tương thích này , những đặc điểm đó được liệt kê dưới đây .

- Một USART thay vì 2 , chỉ trong chế độ dị bộ . Chỉ có 8 bit có nghĩa nhỏ nhất của thanh ghi Baud Rate là có ích .
- Giao diện 2 dây nối tiếp thì không được hỗ trợ .
- Cổng C chỉ là cổng ra
- Cổng G chỉ phục vụ chức năng xoay chiều ( luân phiên )
- Cổng F phục vụ chỉ như là một đầu vào kĩ thuật số thêm vào đầu vào tương tự tới bộ chuyển đổi ADC
- Bộ tải khởi động ( Boot Loader ) không được hỗ trợ
- Không thể điều chỉnh tần số của bộ hiệu chỉnh dao động kế RC ( oscillator ) bên trong .

- Giao diện bộ nhớ bên ngoài có thể không giải phóng bất cứ chân địa chỉ cho cổng I/O chung , không phải cấu hình các chế độ chờ khác nhau đến các khu vực địa chỉ nhớ bên ngoài .

Thêm vào đó , có một vài điểm khác biệt nhỏ để làm nên khả năng tương thích với Atmega 103

- chỉ EXTRF và PORF ra trong MCUCSR
- kết quả thời gian thì không cần thiết cho timer vWatch dog chuyển đổi thời gian chờ
- chân ngắn ngoài 3 – 0 phục vụ chỉ các mức ngắn
- USART không có bộ đệm FIFO , nhưng dữ liệu vẫn vượt qua đến sớm hơn
- Những bít I/O không sử dụng ở trong Atmega 103 nên được viết là O để bảo đảm rằng hoạt động giống như Atmega 128

### Mô tả ý nghĩa các chân ( Pin descriptions )

- VCC : chân cấp nguồn
- GND : Chân nối đất
- Port A (PA7...PA0) : Cổng A là một cổng vào ra hai hướng 8 bit với điện trở hâm ở bên trong (được lựa chọn cho mỗi bit ). Bộ đệm đầu ra của cổng A có đặc tính đối xứng với cả 2 tản nhiệt nguồn cấp .
- Port B (PB7...PB0) : cổng B là một cổng vào ra 2 hướng với điện trở hâm (lựa chọn cho mỗi bit ). Bộ đệm cổng B có tính đối xứng với 2 tản nhiệt và nguồn cấp .
- Port C (PC7...PC0) : cổng C là một cổng vào ra 2 hướng . Bộ đệm đầu ra của cổng C có tính đối xứng
- Port D (PD7...PD0) : cổng D tương tự như cổng D
- Port E tương tự như cổng E
- Port F : trợ giúp giông như những cổng vào tương tự analog cho bộ chuyển đổi A/D . Cổng F cũng là một cổng vào ra 2 hướng nếu như bộ chuyển đổi A/D không được sử dụng . Các chân của cổng này có các trở kháng hâm được lựa chọn cho mỗi bit. Chân TDO là chân có 3 chế độ trừ khi chế độ TAP xuất tín hiệu ra được bật.. Cổng F cũng trợ giúp chức năng của giao diện JTAG
- Port G (PG4...PG0) : cổng G là một cổng vào ra 5 bit 2 hướng với điện trở hâm (được lựa chọn cho từng bit ). Bộ đệm cổng G có tính đối xứng với tản nhiệt và nguồn cấp . Cổng G cũng cung cấp những tính năng đặc biệt . Các chân của cổng G là các cổng có 3 chế độ khi mà điều kiện reset được kích hoạt dù là đồng hồ không chạy
- RESET : đầu ra reset . Cấp cho phép trên chân này thì dài hơn độ dài xung tối thiểu sẽ phát ra tín hiệu reset , cho dù đồng hồ không chạy .
- XTAL1 : đầu vào bộ khuyếch đại dao động và đầu vào cho các đồng hồ đếm bên trong mạch điện điều khiển

- XTAL2 đầu ra cho bộ khuỷu tần đại dao động
- AVCC : là chân nguồn áp cấp cho cổng F và các bộ chuyển đổi A/D . Nó nên là chân nối với VCC , dù là ADC không được sử dụng . Nếu ADC được sử dụng , nó nên được nối với chân VCC thông qua 1 bộ lọc thấp tần
- AREF : là chân tham khảo cho bộ chuyển đổi A/D
- PEN : là chân được kích hoạt trình cho kiểu lập trình nối tiếp SPI , và các tín hiệu vào được kéo lên cao . Bằng việc giữ chân này ở mức thấp trong suốt quá trình khởi động lại nguồn ( Power – on Reset ) , thiết bị này sẽ nhập vào cổng lập trình nối tiếp SPI . PEN không có chức năng gì trong quá trình điều khiển .

## Sự duy trì dữ liệu

Kết quả của sự thẩm định độ bền chỉ ra rằng tốc độ hỏng dữ liệu thì nhỏ hơn 1 PPM trên 20 năm ở nhiệt độ 85 độ C hoặc 100 năm ở 25 độ C

Về các ví dụ mẫu : datasheet này bao gồm các ví dụ code mẫu theo một cách ngắn gọn chỉ ra cách sử dụng các phần khác nhau của thiết bị này. Các đoạn code mẫu này giả thiết rằng các phần xác định tiêu đề của file thì được cài đặt sẵn trước khi được biên dịch . Để nhận biết rằng không phải tất cả các trình biên dịch C được cung cấp bao gồm các bit được xác định ở tiêu đề của file và các quá trình ngắn trong C thì phụ thuộc vào trình biên dịch .xem thêm trong tài liệu của trình biên dịch C để biết thêm chi tiết.

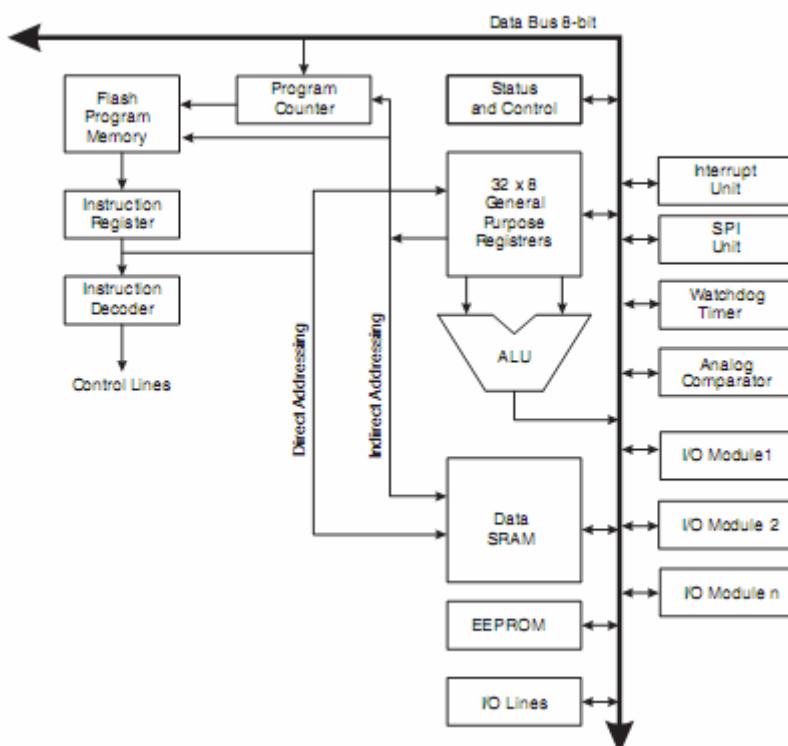
Sự xác định các vị trí của các thanh ghi vào ra trong bản đồ I/O , IN , OUT , SBIC , CBI, và SBI các lệnh phải được thay thế với các lệnh được cho phép truy cập trong phần I/O mở rộng . Thông thường , LDS và STS được kết hợp với SBRS , SBRC, SBR , CBR.

### III .AVR CPU core : Lõi CPU của AVR

Giới thiệu :

Phần này giới thiệu về cấu trúc chung của lõi AVR. Chức năng chính của lõi CPU là để đảm bảo thực hiện đúng chương trình. CPU vì vậy phải có thể truy cập, quản lý bộ nhớ, tiến hành tính toán, điều khiển ngoại vi và xử lý các ngắt.

Tổng quan cấu trúc : hình 3 là sơ đồ khái niệm của AVR



Để có được hiệu năng cao nhất và khả năng làm việc song song, AVR sử dụng cấu trúc Harvard – với sự phân chia bộ nhớ và các bus cho chương trình và dữ liệu. Các lệnh trong bộ nhớ chương trình thì được thực thi với 1 cấp sử lý liên lệnh đơn. Trong khi lệnh đang được xử lý thì lệnh tiếp theo được nạp tiếp từ bộ nhớ chương trình. Khái niệm này kích hoạt lệnh để thực thi trong mỗi chu kỳ xung nhịp đồng hồ. Bộ nhớ chương trình là bộ nhớ flash có thể lập trình lại được ở trong hệ thống.

Sự truy cập nhanh vào file của thanh ghi thì bao gồm 32\*8 bit thanh ghi đa năng với 1 chu kỳ xung nhịp để quản lý thời gian. Điều này cho phép điều khiển trong một chu kỳ đơn của đơn vị sử lý số học ALU. Thông thường trong hoạt động của ALU, 2 toán hạng địa chỉ được xuất ra từ file thanh ghi, quá trình điều khiển được thực thi và kết quả được lưu trữ lại trong thanh ghi file – trong mỗi chu kỳ xung nhịp.

6 trong 32 thanh ghi có thể được sử dụng như là 3 địa chỉ 16 bit gián tiếp cho vùng dữ liệu địa chỉ - kích hoạt địa chỉ có hiệu lực trong tính toán. 1 trong những con trỏ địa chỉ này có thể được sử dụng như là một con trỏ địa chỉ cho việc tìm kiếm các bảng trong bộ nhớ chương trình Flash. Các thanh ghi chức năng được thêm vào là thanh ghi 16 bit thanh ghi X, Y, Z sẽ được miêu tả sau trong phần này.

Đơn vị xử lý số học và logic ALU hỗ trợ quá trình điều khiển số học và logic giữa các thanh ghi hoặc giữa các đại lượng không đổi và các thanh ghi . Các thanh ghi điều khiển quá trình đơn có thể cũng được thi hành trong ALU . Sau một quá trình điều khiển số học , trạng thái của các thanh ghi được cập nhật để phản ánh thông tin về kết quả của quá trình điều khiển .

Dòng chương trình thì được cung cấp bởi các lệnh nhảy có điều kiện và không có điều kiện và các lệnh gọi (call instructions ) , có thể là các địa chỉ trực tiếp trong toàn bộ không gian địa chỉ . Hầu hết các lệnh của AVR đều có định dạng là 16 bit từ đơn . Mỗi bộ nhớ địa chỉ chương trình thì bao gồm 16 hoặc 32 bit lệnh

Không gian nhớ Flash được chia ra làm 2 phần , phần chương trình khởi động và phần chương trình ứng dụng . Cả 2 phần này đều có các bit khóa riêng cho sự bảo vệ ghi và đọc/ghi . Lệnh SPM được viết vào trong bộ nhớ ứng dụng Flash phải được thường chú trong khu vực khởi động chương trình .

Trong suốt quá trình ngắn và gọi các chương trình con, sự hoàn trả địa chỉ của bộ đếm chương trình được lưu ở trong ngăn xếp (Stack ) . Ngăn xếp ( stack ) được cách gán hiệu quả trong SRAM dữ liệu chung , và hiệu quả của ngăn xếp ( stack ) thì chỉ bị giới hạn bởi độ lớn của SRAM và sự sử dụng của SRAM . Tất cả các chương trình sử dụng phải được khởi tạo SP trong chương trình con reset ( trước khi chương trình con hoặc các ngắn được thực thi ) . Con trỏ ngăn xếp ( SP- stack pointer ) là quá trình truy cập đọc/ghi ở trong không gian địa chỉ I/O. SRAM dữ liệu có thể dễ dàng được truy cập đến thông qua 5 kiểu địa chỉ khác nhau được hỗ trợ ở trong cấu trúc của AVR .

Không gian nhớ ở trong cấu trúc của AVR thì đều tuyến tính và đều là các vùng nhớ thông thường . Một module ngắn linh hoạt có các thanh ghi điều khiển của nó ở trong không gian I/O với 1 bit ngắn kích hoạt chung được thêm vào ở trong thanh ghi trạng thái . Tất cả các ngắn đều có một véc tơ phân chia ngắn ở trong các bảng vecto ngắn . Các ngắn thì có quyền ưu tiên phù hợp với vị trí các vecto ngắn của chúng . các vecto ngắn mức thấp , các vecto ngắn mức cao được ưu tiên hơn .

Vùng không gian địa chỉ nhớ I/O bao gồm 64 địa chỉ cái mà có thể được truy cập trực tiếp ,hoặc các vị trí lưu dữ liệu theo các thanh ghi từ \$20- \$5F , Thêm vào đó Atmega 128 còn có thêm không gian địa chỉ I/O mở rộng từ \$60 - \$FF ở trong SRAM nơi mà chỉ có các lệnh như ST/STS/STD hoặc LD/LDS/LDD có thể được sử dụng .

## ALU – đơn vị xử lý số học và logic

Hiệu suất cao của đơn vị xử lý logic của AVR điều khiển một cách trực tiếp việc kết nối với tất cả 32 thanh ghi đa năng chung . Trong vòng một chu kỳ xung nhịp đồng hồ , quá trình điều khiển số học giữa các thanh ghi đa năng tổng hợp hoặc giữa các thanh ghi và 1 sự kiện đang được thực thi ngay lúc đó . Quá trình điều khiển ALU thì được chia ra làm 3 nhóm – số học , logic , và bit chức năng ( bit functions ) . Vài cài đặt của cấu trúc cũng cung cấp những sự trợ giúp đa nhiệm mạnh mẽ cho cả 2 loại tín

hiệu/không tín hiệu phép nhân và định dạng phân số . xem thêm phần cài đặt lệnh để được miêu tả chi tiết hơn .

## Thanh ghi trạng thái – status registers

Thanh ghi trạng thái bao gồm những thông tin về kết quả của tất cả các lệnh số học được thực thi gần nhất . Thông tin này có thể được sử dụng cho sự thay đổi các dòng chương trình để mà thực hiện các điều kiện của quá trình điều khiển . Chú ý rằng thanh ghi trạng thái được cập nhật sau quá trình điều khiển ALU như là được xác định ở trong phần tham khảo cài đặt lệnh . điều này sẽ được gỡ bỏ trong nhiều trường hợp khi cần thiết phải sử dụng các lệnh so sánh riêng , kết quả của việc này là ta có các đoạn mã nhanh hơn và chặt chẽ hơn .

Thanh ghi trạng thái không tự động lưu khi nhập vào 1 chương trình con ngắn và sẽ khôi phục khi phản hồi từ 1 ngắn . điều này phải được điều khiển bởi phần mềm

Thanh ghi trạng thái – SREG- được xác định như là :

Bit	7	6	5	4	3	2	1	0	SREG
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

- Bit 7- I Bit ngắt kích hoạt chung:

phải được cài đặt cho các ngắn được kích hoạt . Sự điều khiển kích hoạt các ngắn riêng lẻ sau đó được sử dụng trong 1 thanh ghi điều khiển sự phân chia . Nếu toàn bộ thanh ghi kích hoạt ngắn bị xóa , thì không có bất cứ ngắn nào được kích hoạt độc lập trong số các ngắn riêng rẽ được cài đặt kích hoạt . Bít I bị xóa bằng phần cứng sau khi 1 ngắn gấp sự cố , và được cài đặt bằng lệnh RETI để kích hoạt lại các chương trình con phục vụ ngắn . Bit I cũng có thể được cài đặt hoặc bị xóa trong phần mềm với lệnh SEI và CLI như là được mô tả trong phần tham khảo lệnh cài đặt

- Bít 6 – T : bit sao chép kho dữ liệu

Bít lệnh copy BLD (bit LoaD ) và BST ( bit STore) sử dụng Bit T như là nguồn hoặc đích đến của bit điều khiển . Một bit từ một thanh ghi trong file thanh ghi có thể được sao chép vào trong Bit T bằng lệnh BST , và một bit trong T có thể được sao chép vào trong một bit ở trong thanh ghi của thanh ghi file bằng lệnh BLD

- Bit 5 – H : cờ báo 1 nửa ( half carry flag )

Bít cờ báo 1 nửa H hiển thị 1 nửa số nhớ trong vài quá trình tính toán số học . bit này rất là hữu dụng ở trong đại số BCD ( xem phần mô tả cài đặt lệnh để biết thêm chi tiết )

- Bit 4 – S : bít báo hiệu ( sign bit ) , S = N+V

Bit S thì luôn luôn là riêng biệt hoặc giữa 2 cờ âm N và dòng tràn bổ sung của cờ V . xem phần mô tả lệnh cài đặt để biết thêm chi tiết .

- Bit 3 – V : cờ báo tràn bổ sung của 2

Cờ báo tràn bộ sung 2 V hỗ trợ phần bù số học của 2 . xem phần mô tả lệnh để biết thêm chi tiết

- Bit 2 – N : cờ báo âm

Cờ báo âm N hiển thị 1 kết quả âm trong một quá trình tính toán số học hoặc logic . xem thêm phần mô tả lệnh để biết thêm chi tiết .

- Bit 1 – Z : cờ không ( zero )

Cờ không Z hiển thị một kết quả zéro trong một quá trình tính toán logic hoặc số học . xem phần mô tả cài đặt lệnh để biết thêm chi tiết

- Bit 0 – C cờ mang :

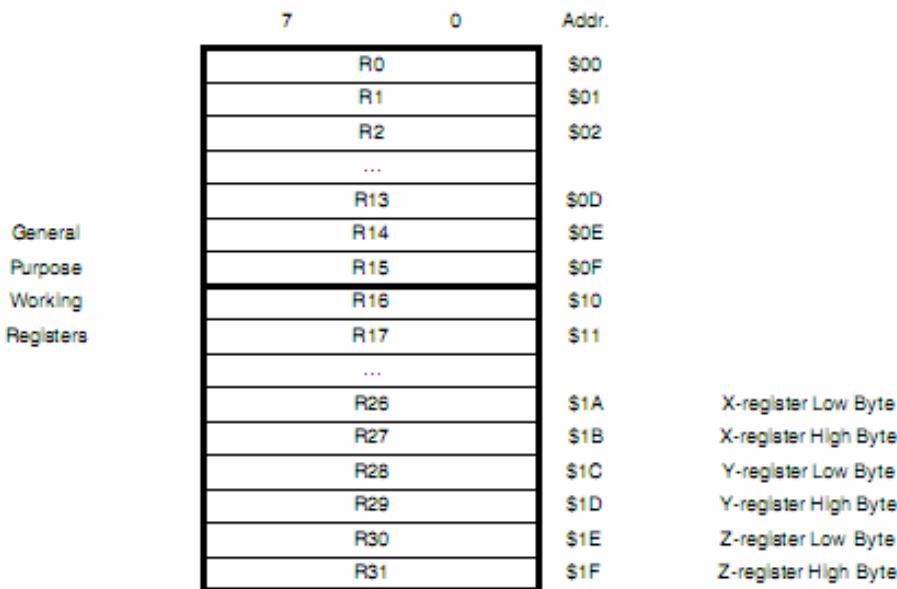
Cờ mang C hiển thị một số mang trong 1 quá trình tính toán logic và số học

### File đăng ký đa năng dùng chung ( general purpose register file )

File đăng ký được tối ưu hóa cho AVR được tăng cường nhờ việc cài đặt lệnh RISC. Để đạt được hiệu suất và độ linh hoạt cần thiết , các giản đồ đầu vào ra (input/output ) sau đây dùng để hỗ trợ file đăng ký :

- 1 toán hạng đầu ra 8 bit và 1 kết quả đầu vào 8 bit
- 2 toán hạng đầu ra 8 bit và 1 kết quả đầu vào 8 bit
- 2 toán hạng đầu ra 8 bit và 1 kết quả đầu vào 16 bit
- 1 toán hạng đầu ra 16 bit và 1 kết quả đầu vào 16bit

Hình 4 chỉ ra cấu trúc của 32 thanh ghi đa năng dùng chung trong CPU

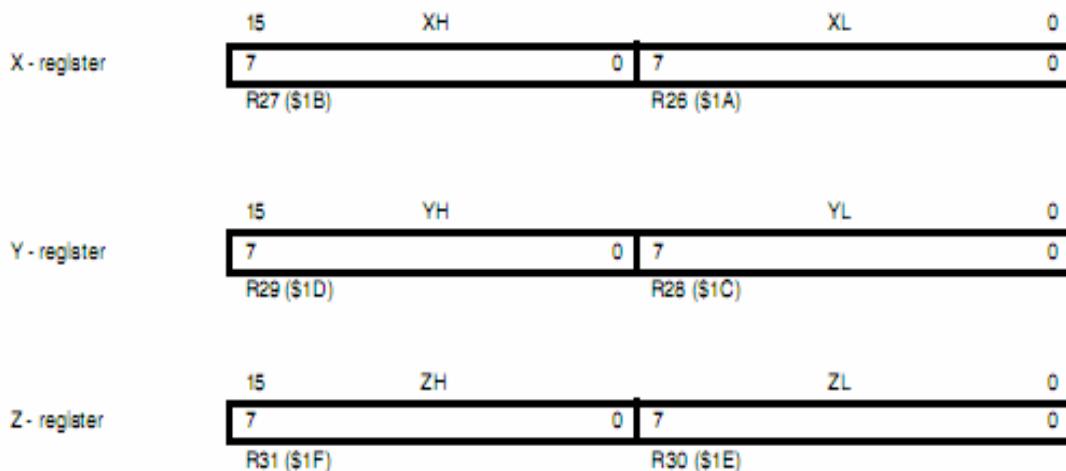


Hầu hết các lệnh điều khiển trong file đăng ký đều có thể truy cập trực tiếp vào tất cả thanh ghi và hầu hết các lệnh trong chúng đều thực hiện trong một chu kỳ xung nhịp.

Như là được chỉ ra trong hình 4 mỗi thanh ghi cũng được gán địa chỉ vùng nhớ dữ liệu , sự sắp xếp trực tiếp vào trong 32 vị trí đầu tiên của không gian dữ liệu người dùng . Mặc dù thiết bị vật lý như là được định vị trong SRAM , sự tổ chức vùng nhớ này cung cấp sự linh hoạt trong truy cập vào các thanh ghi , Như là các con trỏ thanh ghi X, Y ,Z có thể được cài đặt trong bảng của bất cứ thanh ghi nào trong file

## Thanh ghi X , thanh ghi Y , thanh ghi Z

Thanh ghi R26...R31 có một vài chức năng được thêm vào các vùng nhớ đa năng của chúng . Các thanh ghi là các con trỏ địa chỉ 16 bit cho việc đặt địa chỉ một cách gián tiếp trong vùng dữ liệu . Có 3 thanh ghi địa chỉ gián tiếp X, Y , Z thì được miêu tả trong hình 5



Trong các kiểu đặt địa chỉ khác nhau có nhiều thanh ghi địa chỉ có các chức năng như là sự thay thế cố định , tự động gia tăng ,và tự động giảm ( xem thêm phần tham khảo cài đặt lệnh để biết thêm chi tiết )

## Stack pointer : con trỏ ngăn xếp

Ngăn xếp thì được sử dụng chính cho việc lưu trữ dữ liệu tạm thời , cho việc lưu trữ các biến địa phương và việc lưu trữ các địa chỉ phản hồi sau khi gọi các chương trình ngắt và các chương trình con. Thanh ghi con trỏ ngăn xếp luôn luôn ghi ở trên đỉnh của ngăn xếp (Stack ). Chú ý rằng Ngăn xếp được cài đặt như là phát triển từ những vị trí nhớ cao hơn đến các vị trí nhớ thấp hơn .Điều này gợi ý rằng Ngăn xếp đầy các lệnh đã được rút ngắn xuống con trỏ ngăn xếp

Con trỏ ngăn xếp chỉ vào ngăn xếp dữ liệu SRAM nơi mà các chương trình con và các ngăn xếp ngắt được đặt . Không gian ngăn xếp trong SRAM phải được xác định bằng chương trình trước khi bắt cú lệnh gọi chương trình con nào được thực thi hoặc là các ngắt được kích hoạt . Con trỏ ngăn xếp phải được cài đặt ở điểm trên \$60 . Con trỏ ngăn xếp bị suy giảm bằng 1 khi dữ liệu bị đẩy lên ngăn xếp khi dùng lệnh PUSH , và suy giảm bằng 2 khi sự phản hồi địa chỉ bị đẩy vào trong ngăn xếp với sự gọi các chương trình con hoặc các ngắt. Con trỏ ngăn xếp tăng lên bằng 1 khi dữ liệu bị tràn ra khỏi ngăn xếp với lệnh POP , và nó tăng lên 2 khi dữ liệu bị tràn khỏi ngăn xếp với sự phản hồi từ chương trình con RET hoặc phản hồi từ ngắt RETI

Con trỏ ngăn xếp của AVR được cài đặt như là 2 thanh ghi 8 bit trong không gian vào ra I/O . Số các bit thực sự được dùng thì phụ thuộc vào sự cài đặt trước . Chú ý

rằng không gian dữ liệu trong một vài sự cài đặt trước của cấu trúc của AVR thì nhỏ đến nỗi mà chỉ cần SPL. Trong trường hợp này, thanh ghi SPH sẽ không được trình bày

Bit	15	14	13	12	11	10	9	8	SPH
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPL
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

### Thanh ghi lựa chọn RAM page Z - RAMPZ

Bit	7	6	5	4	3	2	1	0	RAMPZ0	RAMPZ
	-	-	-	-	-	-	-	RAMPZ0		RAMPZ
Read/Write	R	R	R	R	R	R	R	R/W		
Initial Value	0	0	0	0	0	0	0	0		

- Bit 7...1 – RES : bit dự trữ ( Reserved Bits )

Có các bít dự trữ và sẽ luôn luôn được đọc là 0 . Khi viết vào các vị trí địa chỉ này viết những bit đó là không cho sự tương thích với các thiết bị trong tương lai.

- Bit 0 – RAM PZ0 : extended RAM page Z – pointer

Thanh ghi RAMPZ thường được sử dụng để chọn lựa cái mà 64K RAM page được truy cập bằng con trỏ Z . Vì Atmega 128 không hỗ trợ nhiều hơn 64K của bộ nhớ SRAM , thanh ghi này chỉ được sử dụng để lựa chọn cái trang mà trong bộ nhớ chương trình được truy cập vào khi mà lệnh ELMP/SPM được sử dụng. Sự cài đặt khác nhau của Bit RAMPZ0 cho ta các hiệu ứng dưới đây :

- RAMPZ0 = 0 địa chỉ nhớ chương trình là từ \$0000 - \$7FFF ( thấp hơn 64K bytes ) được truy cập bởi ELPM/SPM
- RAMPZ0 = 1 địa chỉ nhớ chương trình từ \$8000 - \$FFFF ( cao hơn 64K bytes) thì được truy cập bởi ELPM/SPM

Chú ý rằng LPM thì không có ảnh hưởng bằng việc cài đặt RAMPZ

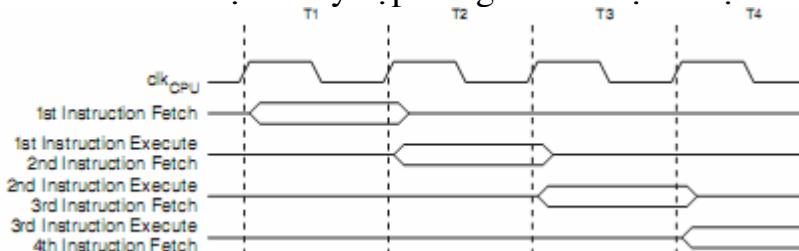
### Lệnh thực thi định thời (instruction execution timing)

Phần này miêu tả khái niệm về sự quản lý truy cập bộ định thời cho sự thực thi các lệnh . CPU của AVR được điều khiển bằng bộ định thời ở trong CPU , nó được sinh ra trực tiếp từ nguồn phát xung đồng hồ đã được chọn đến chip . Không có sự chia bộ đếm thời gian bên nào được sử dụng .

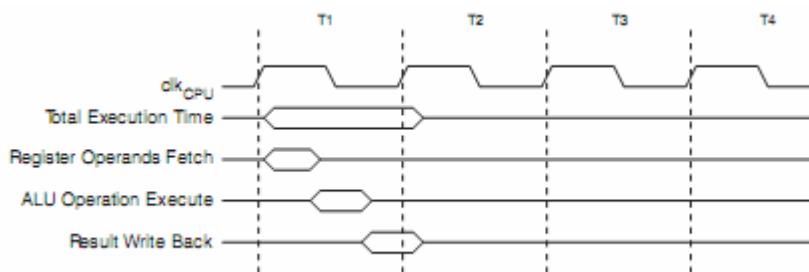
Hình 6 chỉ ra các lệnh truy cập đồng thời và các lệnh thực thi được kích hoạt bằng cấu trúc Harvard và khái niệm file đăng ký truy cập nhanh . Đây là khái niệm xử lý liên lệnh cơ bản thu được trên 1 MIPS trên MHz với kết quả duy nhất tương thích với

cho chức năng trên giá thành , chức năng trên 1 xung nhịp và chức năng trên đơn vị nguồn điện

Hình 6 : lệnh truy cập đồng thời và lệnh thực thi



Hình 7 chỉ ra khái niệm bộ định thời bên trong cho file đăng kí . Trong một chu kì xung nhịp đơn của 1qua trình tính toán ALU thì có 2 thanh ghi được thực thi , và kết quả được lưu lại trong thanh ghi đích đến



Quá trình điều khiển ALU trong một chu kì đơn

### **Khởi động lại và điều khiển ngắn**

AVR cung cấp đa dạng các nguồn ngắn khác nhau . Các ngắn này và mỗi vecto phân chia ngắn có một vecto chương trình ngắn ở trong vùng nhớ chương trình . Tất cả các vecto ngắn đều được gán với các bit riêng rẽ cái mà phải được viết là mức logic 1 cùng với các bit kích hoạt ngắn chung trong thanh ghi trạng thái để mà kích hoạt các ngắn . Phụ thuộc vào giá trị của bộ đếm chương trình , các ngắn có thể tự động vô hiệu hóa khi mà bít khóa khởi động BLB02 hoặc BLB12 được lập trình . Đặc điểm cải thiện tính bảo mật của phần mềm . xem thêm phần bộ nhớ chương trình ở trang 286 để biết thêm chi tiết .

Mức địa chỉ thấp nhất trong vùng nhớ chương trình thì được đặt là mặc định như là RESET và các vecto ngắn . Một danh sách đầy đủ của các vecto ngắn được chỉ ra ở trang 60 . Danh sách cũng xác định rõ các cấp ưu tiên của các ngắn khác nhau. Các địa chỉ thấp hơn các địa chỉ cao thì là các cấp ưu tiên . RESET có mức ưu tiên cao nhất và tiếp theo là INT0 – truy vấn ngắn ngoài . Các vec to ngắn có thể được di chuyển để bắt đầu cho vùng Flash khởi động bằng việc cài đặt bit IVSEL trong thanh ghi điều khiển MCU (MCUCR) . Tham khảo phần ngắn ở trang 60 để biết thêm thông tin . Vecto reset có thể cũng được di chuyển để bắt đầu vùng Flash khởi động bằng cách lập trình BOOTRST fuse , xem thêm phần hỗ trợ tải quá trình khởi động trang 273

Khi một ngắt xuất hiện , Bit I kích hoạt ngắt chung bị xóa và tất cả các ngắt bị vô hiệu hóa . Phần mềm người sử dụng có thể viết mức logic 1 vào bit I để kích hoạt khỏi ngắt . Tất cả các ngắt đã kích hoạt có thể ngắt chương trình con phục vụ ngắt hiện hành . Bít I tự động cài đặt khi có một phản hồi từ lệnh ngắt RETI được thực thi .

Đây là cơ sở của 2 loại ngắt . Loại thứ nhất thì được khởi động bằng một sự kiện cái mà cài đặt cờ báo ngắt . Với những ngắt này , bộ đếm chương trình được vecto hóa đến các vecto ngắt thực sự để mà thực thi việc điều khiển các chương trình con phục vụ ngắt , và phần cứng xóa các cờ ngắt tương ứng . Cờ ngắt cũng có thể được xóa bằng việc viết mức logic 1 lên bit vị trí cờ ngắt bị xóa . Nếu 1 điều kiện ngắt xuất hiện trong khi bit kích hoạt ngắt tương ứng bị xóa ,cờ ngắt sẽ cài đặt và được ghi nhớ cho đến khi quá trình ngắt được kích hoạt , hoặc cờ ngắt bị xóa bằng phần mềm . Tương tự , nếu một hoặc nhiều điều kiện ngắt xuất hiện trong khi bit kích hoạt ngắt chung bị xóa, cờ báo ngắt tương ứng sẽ được cài đặt và ghi nhớ cho đến khi bít kích hoạt ngắt chung được cài đặt và sẽ thực thi sau đó bằng thứ tự ưu tiên .

Loại thứ 2 của các ngắt sẽ khởi động chỉ cần điều kiện ngắt được đưa ra . Những ngắt này không cần thiết phải có cờ báo ngắt . Nếu điều kiện ngắt biến mất trước khi các ngắt này được kích hoạt các ngắt này sẽ không khởi động .

Khi AVR thoát ra từ một ngắt , nó sẽ luôn luôn phản hồi từ chương trình chính và thực hiện 1 hoặc nhiều lệnh trước khi bắt cứ ngắt đang trì hoãn nào được xử lý .

Chú ý rằng thanh ghi trạng thái thì không tự động lưu trữ khi nhập vào một chương trình con phục vụ ngắt hoặc được khôi phục lại khi sự phản hồi từ một chương trình con phục vụ ngắt . Điều này phải được điều khiển bằng phần mềm .

Khi sử dụng lệnh CLI để làm biến mất các ngắt , các ngắt sẽ biến mất ngay lập tức . không có ngắt nào sẽ được thực thi sau khi có lệnh CLI , dù là nó xuất hiện đồng thời với lệnh CLI . Ví dụ dưới đây chỉ ra cách này có thể sử dụng để tránh các ngắt trong suốt dãy ghi thời gian của EEPROM

#### Assembly Code Example

```
in r16, SREG      ; store SREG value
cli    ; disable interrupts during timed sequence
sbi EECR, EEMWE  ; start EEPROM write
sbi EECR, EEWE
out SREG, r16     ; restore SREG value (I-bit)
```

#### C Code Example

```
char cSREG;
cSREG = SREG; /* store SREG value */
/* disable interrupts during timed sequence */
_disable_interrupt();
EECR |= (1<<EEMWE); /* start EEPROM write */
EECR |= (1<<EEWE);
SREG = cSREG; /* restore SREG value (I-bit) */
```

Khi việc sử dụng lệnh SEI để kích hoạt các ngắt , các lệnh SEI dưới đây sẽ thực thi trước khi bắt cứ ngắt bị trì hoãn như là được chỉ trong ví dụ này

Assembly Code Example
<pre>sei ; set global interrupt enable sleep; enter sleep, waiting for interrupt ; note: will enter sleep before any pending ; interrupt(s)</pre>
C Code Example
<pre>_enable_interrupt(); /* set global interrupt enable */ _sleep(); /* enter sleep, waiting for interrupt */ /* note: will enter sleep before any pending interrupt(s) */</pre>

## Thời gian đáp ứng các ngắt

Đáp ứng các ngắt thực thi cho tất cả các ngắt của AVR tối thiểu trong 4 chu kỳ xung nhịp đồng hồ . Sau 4 xung nhịp đồng hồ , vec to địa chỉ chương trình của các chương trình con phục vụ ngắt được thực thi . Trong suốt 4 chu kỳ xung nhịp này , bộ đếm chương trình bị đẩy vào trong ngăn xếp . Vecto này thường được nhảy trong các chương trình con phục vụ ngắt , và các lệnh nhảy này tạo ra 3 chu kỳ xung nhịp. Nếu 1 ngắt xuất hiện trong suốt quá trình thực thi của một lệnh nhiều chu kỳ , lệnh này sẽ hoàn thành sau trước khi ngắt được xử lí . Nếu 1 ngắt xuất hiện khi MCU ở trong chế độ ngủ Sleep mode , thời gian đáp ứng thực thi ngắt thì được gia tăng bằng 4 chu kỳ xung nhịp . Sự gia tăng này dẫn đến thêm vào thời gian khởi động từ quá trình ngủ lựa chọn Sleep mode

Một sự phản hồi từ việc điều khiển chương trình con phục vụ ngắt tạo ra 4 chu kỳ xung nhịp . Trong suốt 4 chu kỳ xung nhịp, bộ đếm chương trình ( 2 Bytes) được tràn ra từ ngăn xếp , con trỏ ngăn xếp được gia tăng bằng 2 ,và Bit I ở trong SREG được cài đặt .

## IV . Các bộ nhớ của AVR Atmega 128

Phần này miêu tả các bộ nhớ khác nhau ở trong Atmega 128. Cấu trúc AVR có 2 không gian nhớ chính , bộ nhớ dữ liệu và bộ nhớ chương trình . Thêm vào đó , đặc điểm của Atmega 128 là một bộ nhớ EEPROM cho kho lưu trữ dữ liệu . Tất cả 3 vùng nhớ thì đều dài và ổn định

### Bộ nhớ chương trình flash co thể lập trình lại trong hệ thống

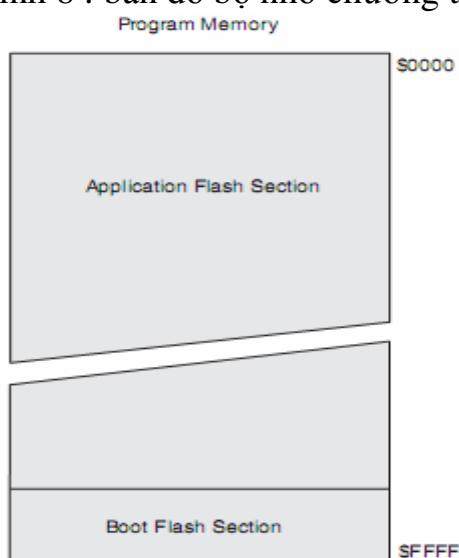
Atmega 128 bao gồm 128K bytes bộ nhớ chương trình có thể lập trình lại trên chip dùng để lưu trữ chương trình . Từ khi tất cả các lệnh của AVR có độ rộng là 16 và 32 bit , bộ nhớ Flash được tổ chức như là 64K\*16 .Để bảo mật phần mềm , không gian bộ nhớ chương trình Flash được chia thành 2 phần , là phần chương trình khởi động và phần chương trình ứng dụng

Bộ nhớ Flash có một độ bền lâu là trên 10000 chu kỳ ghi xóa . Bộ đếm chương trình (PC )của Atmega 128 là 16 bit dài , việc đặt địa chỉ này cho 64K được định vị trong bộ nhớ chương trình . Hoạt động của khu vực chương trình khởi động còn được kết hợp với các bit khóa quá trình khởi động vì sự bảo vệ phần mềm được mô tả một cách chi tiết ở trong phần hỗ trợ tải quá trình khởi động ở trang 273 và lập trình bộ nhớ ở trang 286. ở đó bao gồm những mô tả chi tiết về lập trình cho bộ nhớ Flash trong SPI , JTAG , hoặc kiểu lập trình song song.

Bảng hàng số có thể được gán bên trong không gian địa chỉ bộ nhớ chương trình( xem thêm LPM – load program memory và ELPMP – Extended load program Memory instruction description )

Giản đồ thời gian cho việc cài đặt lệnh và thực thi lệnh được giới thiệu trong phần lệnh thực thi thời gian ở trang 14 .

Hình 8 : bản đồ bộ nhớ chương trình



## Bộ nhớ dữ liệu SRAM : SRAM Data Memory

Atmega 128 hỗ trợ 2 cấu hình khác nhau cho bộ nhớ dữ liệu SRAM như được liệt kê trong bảng 1

Configuration	Internal SRAM Data Memory	External SRAM Data Memory
Normal mode	4096	up to 64K
ATmega103 Compatibility mode	4000	up to 64K

Hình 9 : chỉ ra cách mà bộ nhớ SRAM của Atmega 128 được tổ chức

Atmega 128 là 1 vi xử lý linh hoạt với rất nhiều đơn vị ngoại vi hơn nên có thể hỗ trợ 64 vị trí dự trữ ở trong mã hoạt động của các lệnh IN và OUT . Vì không gian địa chỉ I/O mở rộng từ \$60 đến \$FF trong SRAM , chỉ các lệnh ST/STS/STD và LD/LDS/LDD mới có thể được sử dụng . Không gian địa chỉ I/O không thể xuất ra khi mà Atmega 128 ở trong trạng thái tương thích với Atmega 103

Trong chế độ thông thường địa chỉ vị trí dữ liệu đầu tiên 4352 ở cả hai file đăng kí , bộ nhớ đầu vào ra I/O và dữ liệu trong SRAM . 32 vị trí địa chỉ đầu tiên của thanh ghi file , tiếp theo là 64 vị trí bộ nhớ I/O tiêu chuẩn , sau đó là 160 vị trí của các vùng nhớ I/O mở rộng và tiếp theo là 4096 vị trí địa chỉ của SRAM dữ liệu .

Trong chế độ tương thích với Atmega 103 , đầu tiên là 4096 vị trí địa chỉ vùng dữ liệu ở cả hai file đăng kí , vùng nhớ I/O và SRAM dữ liệu bên trong . Đầu tiên là 32 vị trí địa chỉ của file đăng kí , tiếp theo là 64 vị trí của vùng nhớ I/O chuẩn , và tiếp theo là 4000 vị trí địa chỉ của SRAM bên trong .

Một tùy chọn SRAM dữ liệu bên ngoài nữa có thể được sử dụng với Atmega 128 . SRAM này sẽ chiếm một vùng trong vung địa chỉ còn lại của không gian địa chỉ 64K. Vùng này bắt đầu ở địa chỉ bên dưới trong SRAM . Thanh ghi file , I/O , I/O mở rộng và SRAM trong chiếm các bit thấp nhất 4352 bytes ở chế độ bình thường , và chiếm 4096bytes thấp nhất ở trong chế độ tương thích với Atmega 103 (I/O mở rộng không được đề cập ở đây ) , vì vậy khi sử dụng 64KB(65536 Bytes ) của bộ nhớ ngoài , 61184 Bytes của bộ nhớ ngoài sẽ dư trong chế độ bình thường , và 61440 Bytes trong chế độ tương thích với Atmega 103. xem phần giao diện bộ nhớ bên ngoài ở trang 26 để thêm chi tiết

Khi sự truy cập địa chỉ trong bộ nhớ SRAM vượt quá vị trí bộ nhớ dữ liệu bên trong , SRAM dữ liệu bên ngoài được truy cập sử dụng các lệnh giống nhau về phần truy cập bộ nhớ dữ liệu bên trong . Khi các bộ nhớ dữ liệu bên trong được truy cập , các chân phân tích quá trình đọc và ghi (PG0 và PG1) thì không hoạt động trong khi tất cả truy cập 1 chu kỳ . Quá trình điều khiển SRAM ngoài thì được kích hoạt bằng việc cài đặt các bit SRE trong thanh ghi MCUCR

Sự truy cập SRAM ngoài tạo ra 1 chu kỳ xung nhịp thêm vào trên 1byte được so sánh với sự truy cập vào SRAM trong . Có nghĩa là các lệnh LD , ST , LDS , STS , LDD , STD , PUSH, và POP tạo thêm 1 chu kỳ xung nhịp nữa . Nếu ngăn xếp được đặt

trong SRAM ngoài, các ngắn , sự gọi các chương trình con , và các phản hồi tạo ra 3 xung nhịp bổ xung , bởi vì bộ đếm chương trình 2 bytes bị đẩy lên và tràn ra , và việc truy cập vào bộ nhớ ngoài không tạo ra sự thuận lợi cho sự truy cập bộ nhớ pipe-line bên trong . Khi giao diện SRAM ngoài được sử dụng với trạng thái chờ ( wait- state ) , sự truy cập 1 byte ngoài tạo ra 2 , 3 , 4 xung nhịp thêm vào cho 1 , 2 , 3 trạng thái chờ tương ứng . Các ngắn , các chương trình con , và phản hồi sẽ cần 5 , 7 , 9 xung nhịp nhiều hơn là được xác định ở trong hướng dẫn cài đặt lệnh cho 1 , 2 , 3 trạng thái chờ .

Năm kiểu đặt địa chỉ cho bộ nhớ dữ liệu bao gồm : trực tiếp , gián tiếp kèm thay thế , gián tiếp , gián tiếp kèm tiền giảm bớt , và gián tiếp kèm post – increment . Trong Bộ ghi file , thanh ghi R26 và R31 có đặc điểm là thanh ghi con trả địa chỉ gián tiếp .

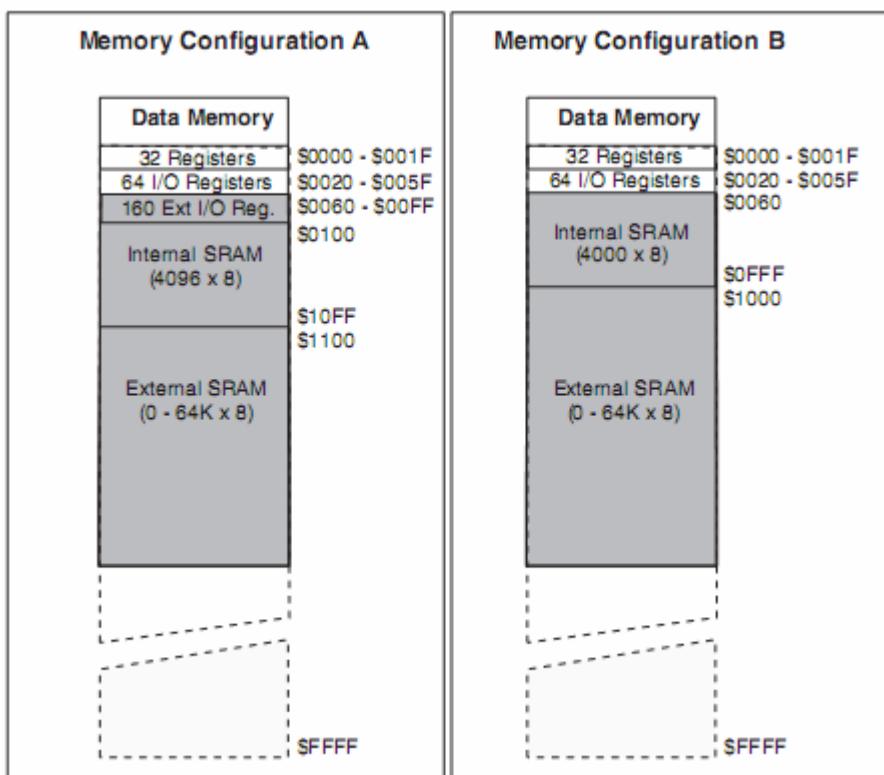
Cách đánh địa chỉ trực tiếp hướng tới không gian địa chỉ trọn vẹn

Cách đánh địa chỉ gián tiếp kèm thay thế hướng tới 63 vị trí địa chỉ từ các đại chỉ cơ sở được đưa ra bởi thanh ghi Y và Z

Khi sử dụng kiểu đánh địa chỉ thanh ghi gián tiếp kèm theo tự động thay thế và post – increment , địa chỉ thanh ghi X , Y , Z được gia tăng hoặc giảm bớt

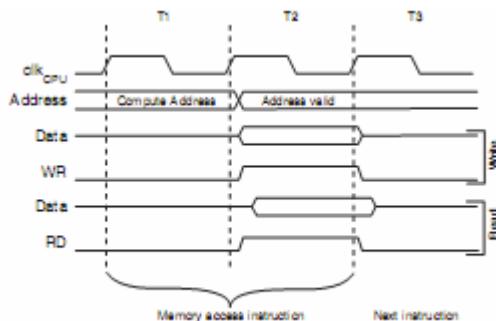
32 thanh ghi làm việc đa năng chung , 64 thanh ghi I/O và 4096 Bytes của SRAM dữ liệu ngoài ở trong Atmega 128 đều được truy cập thông qua tất cả các kiểu đặt địa chỉ trên . bộ ghi file được miêu tả trong “bộ ghi file đa năng dùng chung trang 12 “

Hình 9 bản đồ bộ nhớ dữ liệu



## Thời gian truy cập dữ liệu bộ nhớ : Data memory Access Times

Phần này miêu tả khái niệm thời gian truy cập nói chung của sự truy cập bộ nhớ trong . Sự truy cập bộ nhớ dữ liệu trong SRAM được biểu thị trong 2 xung  $ck_{CPU}$  như là được miêu tả trong hình 10



## Bộ nhớ dữ liệu EEPROM

Atmega 128 bao gồm 4K bytes của bộ nhớ dữ liệu EEPROM . Nó được tổ chức như là sự phân chia không gian dữ liệu , trong đó mỗi bit đơn có thể được đọc và ghi . Bộ nhớ EEPROM có độ bền là trên 100000 chu kỳ ghi xóa . Quá trình truy cập giữa EEPROM và CPU được miêu tả dưới đây , việc xác định địa chỉ thanh ghi địa chỉ EEPROM , thanh ghi dữ liệu EEPROM , và các thanh ghi điều khiển EEPROM

Lập trình bộ nhớ trang 286 bao gồm những miêu tả chi tiết về lập trình EEPROM trong SPI , JTAG , hoặc kiểu lập trình đồng thời .

## Truy cập đọc và ghi của EEPROM

Các thanh ghi quản lý việc truy cập EEPROM được quản lý trong không gian I/O .

Thời gian truy cập ghi cho EEPROM được đưa ra trong Bảng 2 . chức năng tự định thời , tuy nhiên , phần mềm người dùng tự động khi byte kế tiếp được ghi . Nếu mà mã người dùng bao gồm các lệnh mà được viết vào EEPROM , thì một vài sự phồng ngùa phải được đưa ra . Trong bộ nguồn cung cấp đã được lọc kỹ ,  $V_{CC}$  có khả năng tăng hoặc giảm chậm khi bật hoặc tắt nguồn . Vì nguyên nhân này nên thiết bị trong vài chu trình thời gian để chạy ở 1 điện áp thấp hơn đã được xác định vì tần số xung nhịp đã được sử dụng . Xem thêm phần giới thiệu sự sai hỏng EEPROM ở trang 25 để biết thêm chi tiết để tránh các vấn đề có thể xảy ra trong các trường hợp đó.

Để mà tránh sự ngăn cản vô tình EEPROM viết , 1 biện pháp viết xác định phải được tuân theo . Tham khảo phần miêu tả về thanh ghi điều khiển EEPROM để biết thêm chi tiết .

Khi mà EEPROM được đọc , CPU tạm dừng trong khoảng 4 chu kỳ xung nhịp trước khi lệnh được thực thi

## Thanh ghi địa chỉ EEPROM – EEARH và EEARL

Bit	15	14	13	12	11	10	9	8	
	-	-	-	-	EEAR11	EEAR10	EEAR9	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	X	X	X	X	
	X	X	X	X	X	X	X	X	

Bít 15...12 – Res : bít dự trữ

Đây là các bít và luôn luôn được đọc là 0 . Khi viết đến các vị trí địa chỉ này , viết các bit này là 0 để tương thích với các thiết bị trong tương lai

Bít 11...0 – EEAR11..0 : địa chỉ EEPROM

Thanh ghi địa chỉ EEPROM – EEARH và EEARL – xác định các địa chỉ của EEPROM trong không gian 4K bytes EEPROM . Bytes dữ liệu của EEPROM được đánh địa chỉ thuộc trong khoảng giữa 0 và 4096. Giá trị ban đầu của EEAR thì không được xác định . Giá trị chính xác phải được viết trước khi EEPROM có thể được truy cập .

## Thanh ghi dữ liệu EEPROM – EEDR

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	EEDR
Read/Write	R/W								

Bít 7...0 – EEDR7..0 . dữ liệu EEPROM

Để phục vụ cho quá trình ghi EEPROM , thanh ghi EEDR bao gồm dữ liệu được viết vào EEPROM ở các địa chỉ được đưa bởi thanh ghi EEAR . Để phục vụ quá trình đọc EEPROM , thanh ghi EEDR bao gồm các dữ liệu đưa ra từ EEPROM tại địa chỉ được đưa ra bởi EEAR.

## Thanh ghi điều khiển EEPROM – EECR

Bit	7	6	5	4	3	2	1	0	
	-	-	-	-	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

Bít 7...4 – Res : các bit dự trữ

Các bit này là các bit dự trữ trong Atmega 128 và sẽ luôn luôn được đọc là 0 .

Bít 3 – EERIE : sẵn sàng kích hoạt các ngắt EEPROM

Việc viết EERIE để kích hoạt một ngắt sẵn sàng của EEPROM nếu bít I của SREG được cài đặt . Viết EERIE là 0 để vô hiệu hóa ngắt . thanh ghi ngắt sẵn sàng của EEPROM sinh ra 1 ngắt không đổi khi EEWE bị xóa

Bít 2 – EEMWE : bít kích hoạt viết chính của EEPROM

Bít EEMWE quyết định có cài đặt EEWE là 1 nguyên nhân gây ra EEPROM được ghi dữ liệu vào. Khi mà EEMWE được viết là 1 , việc viết EEWE là 1 trong vòng 4 chu kỳ xung nhịp sẽ viết dữ liệu vào EEPROM ở các địa chỉ đã được lựa chọn . Nếu EEMWE là 0 , việc viết EEWE là 1 sẽ không có hiệu lực . Khi EEMWE đã được viết là 1 bởi phần mềm, thì phần cứng sẽ xóa bit về 0 sau 4 chu kỳ xung nhịp. Xem thêm phần miêu tả của bit EEWE cho 1 quy trình viết vào EEPROM

Bit 1- EEWE : Kích hoạt việc viết EEPROM

Tín hiệu EEWE kích hoạt việc ghi vào EEPROM thì được phân tích ở trong EEPROM. Khi địa chỉ và dữ liệu đã được cài đặt chính xác , bit EEWE phải được cài đặt để viết giá trị vào trong EEPROM . Bit EEMWE phải được cài đặt khi mức logic 1 được viết vào bit EEWE , theo cách khác thì việc viết vào EEPROM không xảy ra . Quy trình dưới đây nên được tuân theo khi ghi vào EEPROM ( thứ tự của bước 3 và 4 là không cần thiết )

1. Đợi cho đến khi EEWE trở thành 0
2. Đợi cho đến khi bit SPMEN trong thanh ghi SPMCSR trở thành 0
3. Viết địa chỉ mới của EEPROM tới EEAR ( lựa chọn )
4. Viết dữ liệu mới của EEPROM tới EEDR ( lựa chọn )
5. Viết mức logic 1 tới bit EEMWE trong khi viết mức logic 0 tới bit EEWE ở thanh ghi EECR
6. Trong 4 chu kỳ xung nhịp sau khi cài đặt EEMWE , viết mức logic 0 vào EEWE

Bộ nhớ EEPROM không thể bị lập trình trong suốt quá trình mà CPU ghi vào trong bộ nhớ Flash . Phần mềm phải kiểm tra xem chương trình trong bộ nhớ Flash được hoàn thiện trước khi khởi đầu 1 lần ghi EEPROM . Bước 2 chỉ thực sự cần thiết nếu như phần mềm bao gồm 1 bộ tải quá trình khởi động (boot loader ) đang cho phép CPU lập trình vào trong bộ nhớ Flash. Nếu như bộ nhớ Flash không bao giờ được cập nhật bởi CPU thì bước 2 có thể được bỏ qua . xem phần hỗ trợ tải quá trình khởi động ở trang 273 để biết thêm chi tiết về sự lập trình khởi động

Cảnh báo : một sự gián đoạn giữa bước 5 và bước 6 sẽ làm hỏng 1 chu kỳ ghi , từ khi bit kích hoạt quá trình ghi chính của EEPROM sẽ bị hết thời gian chờ . Nếu như 1 chương trình con phục vụ ngắt đang truy cập vào đang được ngắt bởi một sự truy cập vào EEPROM khác , thanh ghi EEAR và thanh ghi EEDR sẽ bị sửa đổi , nguyên nhân của quá trình ngắt EEPROM bị hỏng . Nó thì được đề nghị phải có cờ báo ngắt chung được xóa trong suốt 4 bước cuối để tránh các vấn đề trên .

Khi thời gian truy cập viết trôi qua , bit EEWE bị xóa bởi phần cứng . Phần mềm người dùng có thể thăm dò bit này và đợi đến khi là 0 trước khi viết byte kế tiếp . Khi

EEWE đã được cài đặt , CPU bị dừng lại trong 2 chu kì xung nhịp trước khi lệnh tiếp theo được thực thi.

#### Bit 0 – EERE : kích hoạt quá trình đọc EEPROM

Tín hiệu EERE kích hoạt việc đọc EEPROM thì được quá trình đọc phân tích ở trong EEPROM . Khi địa chỉ chính xác được cài đặt trong thanh ghi EEAR , bit EERE phải được viết là mức logic 1 tới bộ kích hoạt trong quá trình đọc EEPROM . Quá trình đọc EEPROM tạo ra một lệnh , và được truy vấn dữ liệu có giá trị ngay lập tức . Khi EEPROM được đọc, CPU bị dừng lại trong 4 chu kì xung nhịp trước khi lệnh kế tiếp được thực thi

Người sử dụng nên thăm dò bit EEWE trước khi bắt đầu quá trình đọc . Nếu 1 quá trình viết đang được tiến hành , nó cũng không thể được đọc bởi EEPROM , và cũng không thay đổi thanh ghi EEAR

Sự điều chỉnh bộ tạo dao động thường được sử dụng trong quá trình truy cập EEPROM . Bảng 2 liệt kê các thời gian chương trình thông thường cho sự truy cập EEPROM từ CPU

Symbol	Number of Calibrated RC Oscillator Cycles <sup>(1)</sup>	Type Programming Time
EEPROM Write (from CPU)	8448	8.5 ms

Note: 1. Uses 1 MHz clock, independent of CKSEL-fuse settings.

Các đoạn code mẫu dưới đây chỉ dùng các hàm của C và ASSEMBLY để viết vào EEPROM.

Các ví dụ này thừa nhận rằng các ngắt đã được điều khiển ( ví dụ bằng cách vô hiệu hóa các ngắt 1 cách chung )vì thế mà không có ngắt nào xuất hiện trong suốt quá trình thực thi các hàm trên . Các ví dụ này cũng thừa nhận rằng không có bộ tải quá trình khởi động nào bị ngăn cản trong phần mềm . Nếu như đoạn code bị ngăn cản , thì hàm viết EEPROM cũng phải đợi trong khi bắt cứ lệnh SPM nào đang diễn ra hoàn thành

Assembly Code Example
<pre> EEPROM_write:     ; Wait for completion of previous write     sbic EECR,EEWE     rjmp EEPROM_write     ; Set up address (r18:r17) in address register     out EEARH, r18     out EEARL, r17     ; Write data (r16) to data register     out EEDR,r16     ; Write logical one to EEMWE     sbi EECR,EEMWE     ; Start eeprom write by setting EEWE     sbi EECR,EEWE     ret </pre>

**C Code Example**

```

void EEPROM_write(unsigned int uiAddress, unsigned char ucData)
{
    /* Wait for completion of previous write */
    while(EECR & (1<<EEWE))
        ;
    /* Set up address and data registers */
    EEAR = uiAddress;
    EEDR = ucData;
    /* Write logical one to EEMWE */
    EECR |= (1<<EEMWE);
    /* Start eeprom write by setting EEWE */
    EECR |= (1<<EEWE);
}

```

Các ví dụ mẫu kế tiếp chỉ ra các hàm C và assembly dùng cho quá trình đọc EEPROM . Các ví dụ này cũng thừa nhận rằng các ngắt đã được điều khiển vì thế mà không có ngắt nào xuất hiện trong suốt quá trình thực thi của các hàm trên.

**Assembly Code Example**

```

EEPROM_read:
    ; Wait for completion of previous write
    sbic EECR,EEWE
    rjmp EEPROM_read
    ; Set up address (r18:r17) in address register
    out EEARH, r18
    out EEARL, r17
    ; Start eeprom read by writing EERE
    sbi EECR,EERE
    ; Read data from data register
    in  r16,EEDR
    ret

```

**C Code Example**

```

unsigned char EEPROM_read(unsigned int uiAddress)
{
    /* Wait for completion of previous write */
    while(EECR & (1<<EEWE))
        ;
    /* Set up address register */
    EEAR = uiAddress;
    /* Start eeprom read by writing EERE */
    EECR |= (1<<EERE);
    /* Return data from data register */
    return EEDR;
}

```

## Quá trình viết trong suốt chế độ ngủ ngắt điện (EEPROM Write During Power-down Sleep Mode )

Khi đăng nhập vào trong chế độ ngủ ngắt điện trong khi 1 quá trình viết EEPROM đang hoạt động , quá trình viết EEPROM vẫn tiếp tục và sẽ hoàn thành trước thời gian sự truy cập ghi được chuyển tiếp . Tuy nhiên , khi quá trình viết được hoàn thành , bộ tạo dao động vẫn tiếp tục chạy , và như 1 hệ quả , thiết bị này không truy nhập vào chế độ ngủ ngắt nguồn một cách trọn vẹn . Vì vậy nó được đề nghị kiểm tra lại rằng quá trình viết EEPROM đã được hoàn thành trước khi có sự truy nhập vào chế độ ngủ ngắt nguồn .

### Sự ngăn cản việc sai hỏng EEPROM

Trong suốt quá trình  $V_{CC}$  ở mức thấp , dữ liệu của EEPROM có thể bị hư hỏng bởi vì điện áp nguồn cấp thì quá thấp để cho CPU và EEPROM làm việc bình thường . Những kết quả này thì giống như là về phần EEPROM sử dụng các bậc trong bo mạch hệ thống , và giải pháp thiết kế nên được áp dụng .

Một sự hư hỏng dữ liệu của EEPROM có thể bị gây ra trong hai trường hợp khi mà điện áp quá thấp . Đầu tiên , một quá trình ghi liên tục thông thường tới EEPROM cần có 1 điện áp cực tiểu để điều khiển đúng . Thứ 2 , bản thân CPU thực hiện các lệnh không đúng , nếu như nguồn cung cấp quá thấp .

Sự hư hỏng dữ liệu của EEPROM có thể dễ dàng tránh được bằng các khuyến cáo trình bày dưới đây :

Giữ cho hoạt động RESET (low) của AVR trong suốt giai đoạn thiếu điện áp nguồn cấp . Điều này có thể làm được bằng việc kích hoạt bộ dò sụt áp bên trong (BOD) . Nếu 1 tín hiệu RESET xuất hiện trong khi 1 quá trình ghi đang tiến hành , quá trình ghi sẽ được hoàn thành với điều kiện là điện áp nguồn cấp đang bị thiếu hụt .

### Bộ nhớ vào/ra : I/O Memory

Vùng nhớ I/O xác định trong Atmega 128 thì được chỉ ra trong phần “chi tiết thanh ghi “ ở trang 362

Tất cả các cổng vào ra và các ngoại vi của Atmega 128 đều được đặt trong vùng không gian nhớ I/O . Tất cả các vị trí I/O có thể được truy cập bằng lệnh LD/LDS/LDD và ST/STS/STD , sự chuyển dữ liệu giữa 32 thanh ghi đa năng làm việc chung và không gian I/O . Thanh ghi I/O trong phạm vi các cấp địa chỉ \$00 - \$1F đều là các bit có thể truy cập trực tiếp bằng việc sử dụng lệnh SBI và CBI . Trong các thanh ghi đó , giá trị của các bit đơn có thể được kiểm tra bằng cách sử dụng các lệnh SBIS và SBIC . Tham khảo phần cài đặt lệnh để biết thêm chi tiết . Khi sử dụng I/O để xác định lệnh IN và OUT thì các địa chỉ I/O \$00 - \$3F phải được sử dụng . Khi việc đặt địa chỉ thanh ghi cũng như là vùng dữ liệu sự sử dụng các lệnh LD và ST , \$20 phải được thêm vào các địa chỉ đó . Atmega 128 là một vi xử lý linh hoạt với rất nhiều đơn vị ngoại vi cần được

hỗ trợ trong phạm vi 64 vị trí dự trữ trong mã hoạt động (opcode) cho các lệnh IN và OUT. Cho phần vùng nhớ I/O mở rộng từ \$60 - \$FF trong SRAM, chỉ các lệnh ST/STS/STD và LD/LDS/LDD mới có thể được sử dụng. Vùng không gian I/O mở rộng thì được thay thế với các vị trí của SRAM khi mà Atmega 128 ở trong chế độ tương thích với Atmega 103.

Để tương thích với các thiết bị trong tương lai, các bit dự trữ nên được viết là 0 nếu được truy nhập. Các địa chỉ I/O dự trữ không bao giờ nên viết vào.

Một vài trạng thái của cờ báo bị xóa bằng việc viết mức logic 1 lên chúng. Chú ý rằng các lệnh CBI và SBI sẽ điều khiển tất cả các bit trên thanh ghi I/O, sự viết 1 lùi vào trong bất cứ cờ được đọc nào như là đã được cài đặt, do đó có sự xóa các cờ. Các lệnh CBI và SBI chỉ làm việc với các thanh ghi từ \$00 đến \$1F.

Các thanh ghi ngoại vi và các thanh ghi điều khiển ngoại vi thì được diễn giải trong các phần sau.

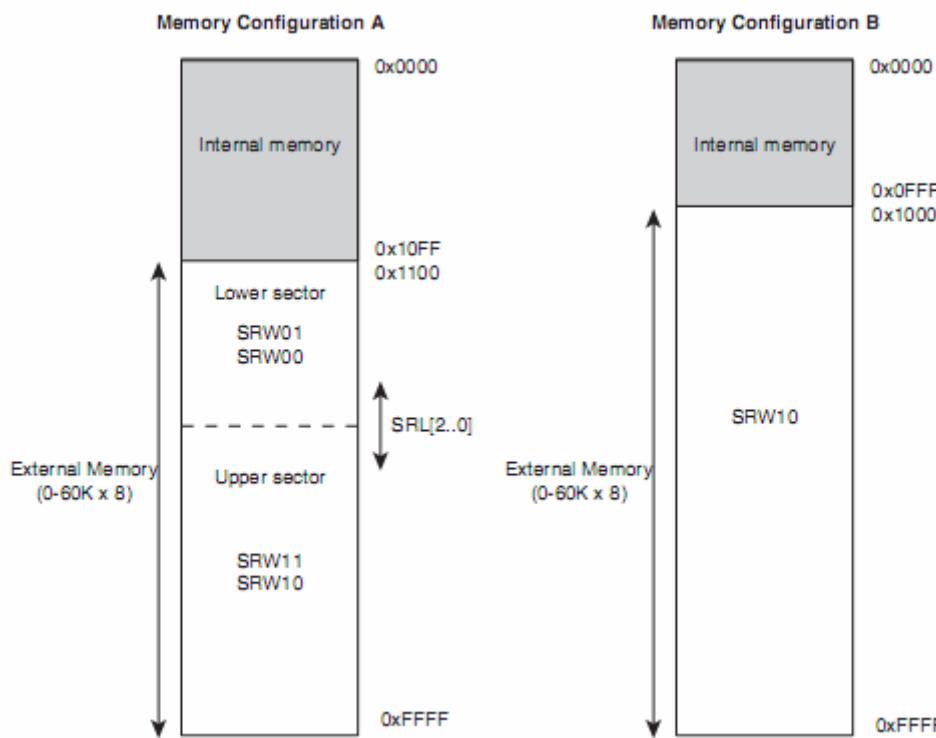
## Giao diện bộ nhớ ngoài : External Memory Interface

Với tất cả các tính năng mà bộ nhớ ngoài cung cấp, nó thì rất phù hợp để điều khiển như là một giao diện đến các bộ nhớ thiết bị như là SRAM ngoài và bộ nhớ Flash, và các ngoại vi như là bộ hiển thị LCD, chuyển đổi A/D, D/A. Các tính năng chính là

- Cài đặt 4 chế độ chờ khác nhau ( bao gồm No wait – state )
- Không phụ thuộc vào việc cài đặt các trạng thái chờ cho các phần bộ nhớ ngoài ( cấu hình các cờ của sector )
- Số của các bit riêng đến các địa chỉ của các bit cao được lựa chọn
- Bộ giữ các bus trên đường dữ liệu tới sự triệt tiêu các dòng điện cực tiểu ( lựa chọn)

## Tổng quan : Overview

Khi các bộ nhớ ngoài (XMEM) được kích hoạt, không gian địa chỉ bên ngoài của SRAM trong trở thành việc sử dụng có ích các chân riêng của bộ nhớ ngoài ( xem hình 1 ở trang 2, bảng 27 ở trang 73 và bảng 33 ở trang 77, bảng 45 ở trang 85. Cấu hình bộ nhớ thì được chỉ ra trong hình 11 >



Chú ý : Atmega 128 không ở trong chế độ tương thích với Atmega 103 : cấu hình bộ nhớ A là khả dụng ( cấu hình bộ nhớ B N/A )

Atmega 128 ở trong chế độ tương thích với Atmega 103 : cấu hình bộ nhớ B là khả dụng ( cấu hình bộ nhớ A N/A)

### Chế độ tương thích với Atmega 103

Cả hai thanh ghi điều khiển bộ nhớ ngoài ( XMCRA và XMCRB ) đều được đặt ở trong không gian I/O mở rộng . Trong chế độ tương thích với Atmega 103 , các thanh ghi này không được khả dụng , và tính năng này được lựa chọn bởi các thanh ghi không khả dụng . Thiết bị này vẫn ở trong chế độ tương thích với Atmega 103 , giống như những tính năng này đã không thoát ra trong Atmega 103 . Sự giới hạn ở trong chế độ tương thích với Atmega 103 là

- chỉ có cài đặt 2 trạng thái chờ ( wait – state ) là khả dụng ( SRW1n = 0b00 và SRW1n = 0b01 )
- Số của các bit mà được truy cập đến các địa chỉ byte cao được sửa chữa .
- Phần bộ nhớ bên ngoài không thể bị phân chia vào trong các phần nhỏ sector với sự cài đặt các trạng thái chờ khác nhau
- Bus – keeper thì không khả dụng
- Các chân RD , WR và ALE chỉ là đầu ra ( cổng G trong Atmega 128 )

## Việc sử dụng giao diện bộ nhớ ngoài : Using the External Memory Interface

Giao diện bao gồm :

- AD7:0 nhiều bus địa chỉ thấp và bus dữ liệu
- A15:8 Bus địa chỉ cao ( có thể cấu hình thứ tự của bit )
- ALE : kích hoạt địa chỉ then chốt
- RD : Phân tích sự đọc
- WR: phân tích sự ghi

Bิต điều khiển của giao diện bộ nhớ ngoài được đặt trong 3 thanh ghi , thanh ghi điều khiển MCU – MCUCR , thanh ghi điều khiển bộ nhớ ngoài A – XMCRA , và thanh ghi điều khiển bộ nhớ bên ngoài B – XMCRB

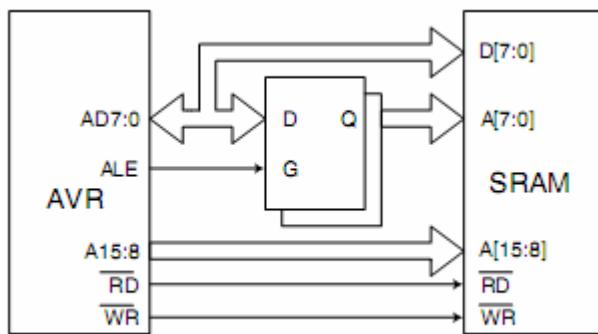
Khi giao diện XMEM được kích hoạt , giao diện XMEM sẽ ghi đè lên quá trình cài đặt ở trong thanh ghi định hướng dữ liệu cái mà tương ứng với các cổng riêng tới giao diện XMEM . Để biết thêm chi tiết về phần cổng ghi đè xem phần chức năng luân phiên của cổng I/O ở trang 86 . Giao diện XMEM sẽ tự động dò một sự truy cập là vào trong hay ra ngoài dù được hay không . Nếu như sự truy cập này là bên ngoài thì giao diện XMEM sẽ xuất địa chỉ ra , dữ liệu , và các tín hiệu điều khiển trên các cổng theo hình 13 ( hình này chỉ ra dạng sóng thiếu trạng thái chờ vWait – state ) . Khi ALE đi từ cao đến thấp có 1 giá trị địa chỉ trên AD7:0. ALE thấp trong suốt quá trình truyền dữ liệu . Khi giao diện XMEM được kích hoạt cũng có 1 sự truy cập vào trong sẽ là gây ra hoạt động trên địa chỉ , dữ liệu , và cổng ALE , nhưng sự phân tích RD và WR không bị dịch chuyển trong suốt quá trình truy nhập vào trong . Khi giao diện bộ nhớ ngoài bị vô hiệu hóa , các chân bình thường và việc cài đặt bit định hướng dữ liệu được sử dụng . Chú ý rằng khi mà giao diện XMEM bị vô hiệu hóa , không gian địa chỉ bên dưới của bộ SRAM nhị phân bên trong không được vẽ vào trong SRAM bên trong . Hình 12 minh họa cách kết nối một SRAM ngoài với AVR bằng việc sử dụng 1 bit then cài 8 bit (an octal latch ) ( thường là “74\*573” hoặc lượng tương đương ) cái mà trong suốt khi cổng G ở mức cao

### Sự cần thiết của Chốt địa chỉ ( Address Latch Requirements )

Do quá trình điều khiển của giao diện XRAM có tốc độ cao , sự chốt địa chỉ phải được lựa chọn cẩn thận cho các tần số của hệ thống dưới 8 MHz @4V và 4MHz @ 2.7V . Khi quá trình điều khiển ở dưới các tần số trên , các series chốt 74HC cũ thông thường trở nên không phù hợp. Giao diện bộ nhớ ngoài được thiết kế để phù hợp với các series chốt 74HC . Tuy nhiên , hầu hết các latch có thể được sử dụng chỉ cần chúng tuân theo các tham số thời gian chính , các tham số thời gian chính cho các chốt địa chỉ là :

- D đến Q lan truyền trễ (  $t_{PD}$  )
- Cài đặt dữ liệu thời gian trước khi G thấp (  $t_{SU}$  )
- Thời gian treo dữ liệu ( địa chỉ ) sau khi G thấp (TH)

Giao diện bộ nhớ ngoài được thiết kế để đảm bảo thời gian treo địa chỉ nhỏ nhất sau khi G được xác nhận ở mức thấp của  $t_h = 5$  ns . Tham khảo thêm  $t_{LAXX\_LD}$  /  $t_{LLAXX\_ST}$  trong phần “biểu đồ bộ nhớ dữ liệu ngoài” bảng 137 đến bảng 144 ở trang 328 -330. Sự lan truyền trễ từ D đến Q (  $t_{PD}$  ) phải được đưa vào xem xét khi mà sự tính toán thời gian truy cập là cần thiết của các thành phần bên ngoài . Thời gian cài đặt dữ liệu trước khi G thấp ( $t_{SU}$  ) phải không được vượt quá giá trị địa chỉ để ALE thấp ( $t_{AVLLC}$  ) giá trị PCB âm của sự trì hoãn quá trình viết ( phụ thuộc vào dung lượng của tải )



Hình 12 : SRAM ngoài được kêt nối với AVR

### Sự dừng lại và bộ giữ Bus ( Pull – up and Bus – keeper )

Các bộ dừng trên cổng AD7:0 có thể được hoạt động nếu thanh ghi cổng tương ứng được viết là 1 . Để giảm lượng tiêu thụ điện ở trong chế độ Sleep mode , nó được khuyến cáo để vô hiệu hóa bộ dừng pull – up bằng việc viết thanh ghi cổng là 0 trước khi đăng nhập vào chế độ ngủ sleep .

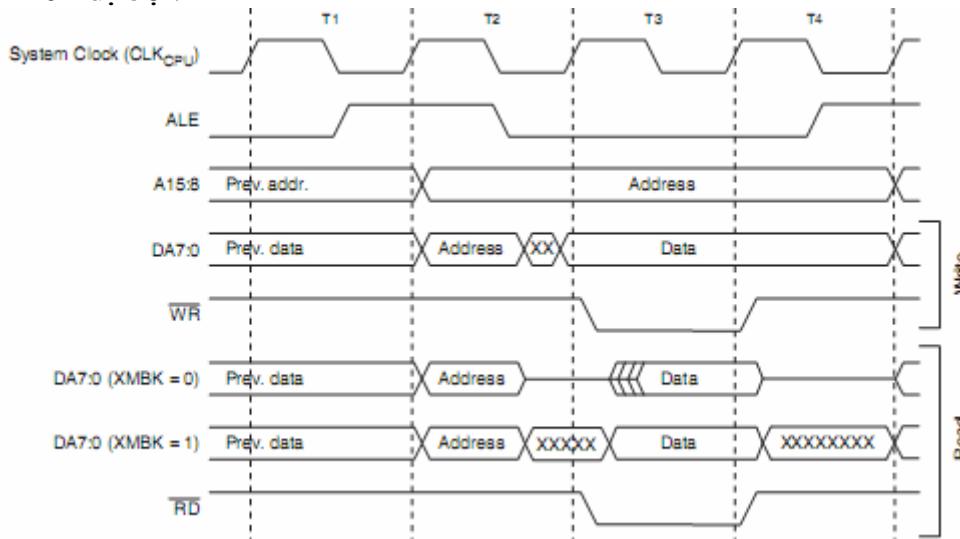
Giao diện XMEM cũng cung cấp 1 bộ giữ Bus – keeper trên các đường AD7:0 . Bộ Bus – keeper này có thể bị vô hiệu hóa và kích hoạt bằng phần mềm , như là được miêu tả trong phần “ External Memory Control Register B – XMCRB ” ở trang 33 . Khi được kích hoạt , bộ giữ bus – keeper sẽ bảo đảm 1 cấp logic xác định (0 hoặc 1 ) ở trên bus AD7:0 khi mà các đường này sẽ khác 3 trạng thái bởi giao diện XMEM

### Bộ định thời

Các thiết bị nhớ bên ngoài có những đòi hỏi các bộ định thời khác nhau . Để đáp ứng các đòi hỏi này giao diện XMEM của Atmega 128 cung cấp 4 trạng thái chờ khác nhau được chỉ ra trong bảng 4 . Nó thì rất quan trọng và được coi như là bảng đặc tính của bộ định thời của các thiết bị nhớ ngoài trước khi quá trình lựa chọn các trạng thái chờ . Các tham số quan trọng nhất là thời gian truy cập cho bộ nhớ bên ngoài được so

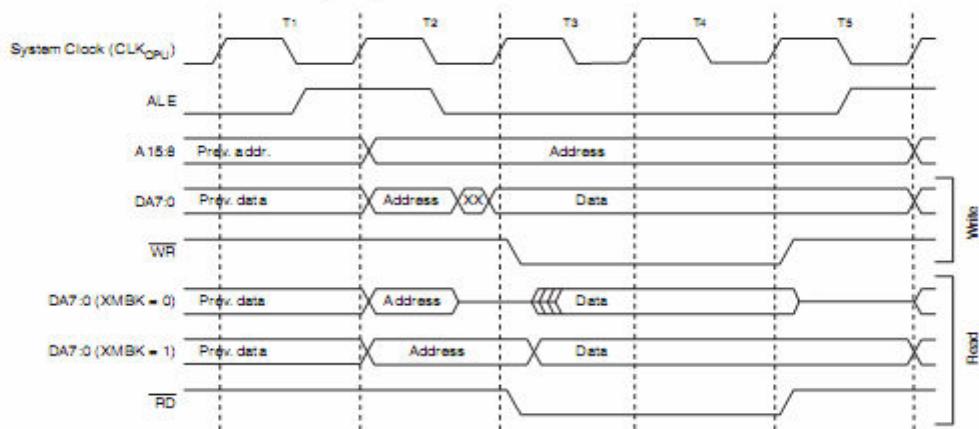
sánh với thủ tục cài đặt của Atmega 128 . Thời gian truy cập bộ nhớ ngoài được xác định bằng thời gian từ lúc chip nhận tín hiệu đến lựa chọn/đánh địa chỉ cho đến khi dữ liệu của các địa chỉ thực này được điều khiển trên Bus . Thời gian truy nhập này không thể vượt quá thời gian từ khi xung ALE phải được xác nhận thấp cho đến khi dữ liệu ổn định trong suốt 1 quá trình đọc liên tục (xem  $t_{LLRL} + t_{RLRH} - t_{DVRH}$  trong bảng 137 – 144 trang 328- 330 ) . Các trạng thái chờ khác nhau được cài đặt bằng phần mềm . Như là một tính năng được thêm vào , nó có thể phân chia bộ nhớ ngoài thành 2 phần nhỏ với sự cài đặt các trạng thái riêng . Điều này làm nó có khả năng kết nối với 2 thiết bị nhớ khác nhau với các yêu cầu về định thời khác nhau giống như giao diện XMEM . Để biết thêm chi tiết về bộ định thời của giao diện XMEM , tham khảo bảng 137 -144 và hình 157-160 trong phần bộ định thời bộ nhớ gắn ngoài ở trang 328

Chú ý rằng giao diện XMEM là dị bộ và dạng sóng trong hình dưới đây thì được liên hệ với đồng hồ xung nhịp bên trong hệ thống . Độ lệch giữa xung nhịp trong và xung nhịp ngoài (XTAL 1 ) thì được bảo đảm ( sự khác nhau giữa nhiệt độ của các thiết bị và nguồn cấp ) . Kết quả của việc này , giao diện XMEM không được thích hợp với quá trình điều khiển dị bộ .



Hình 13 : chu kì xung nhịp bộ nhớ dữ liệu ngoài thiếu chế độ chờ

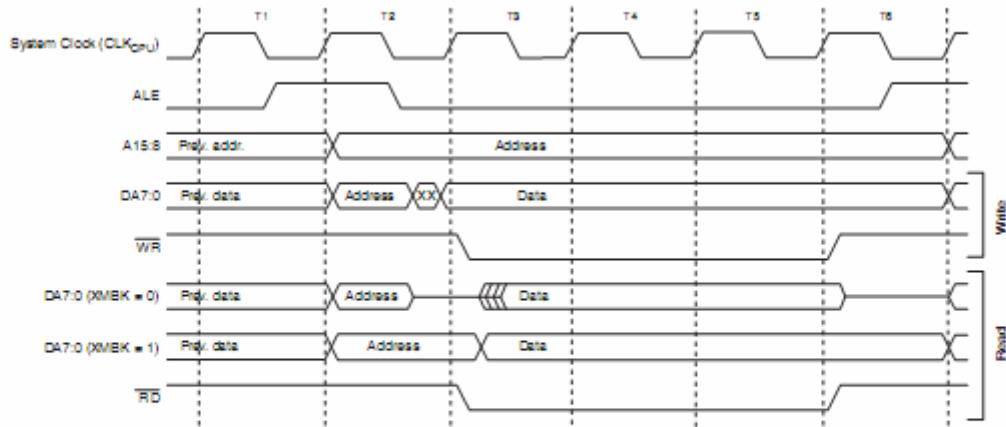
Note: 1. SRWn1 = SRW11 (upper sector) or SRW01 (lower sector), SRWn0 = SRW10 (upper sector) or SRW00 (lower sector). The ALE pulse in period T4 is only present if the next instruction accesses the RAM (internal or external).

**Figure 14.** External Data Memory Cycles with SRWn1 = 0 and SRWn0 = 1<sup>(1)</sup>

Note:

1. SRWn1 = SRW11 (upper sector) or SRW01 (lower sector), SRWn0 = SRW10 (upper sector) or SRW00 (lower sector).

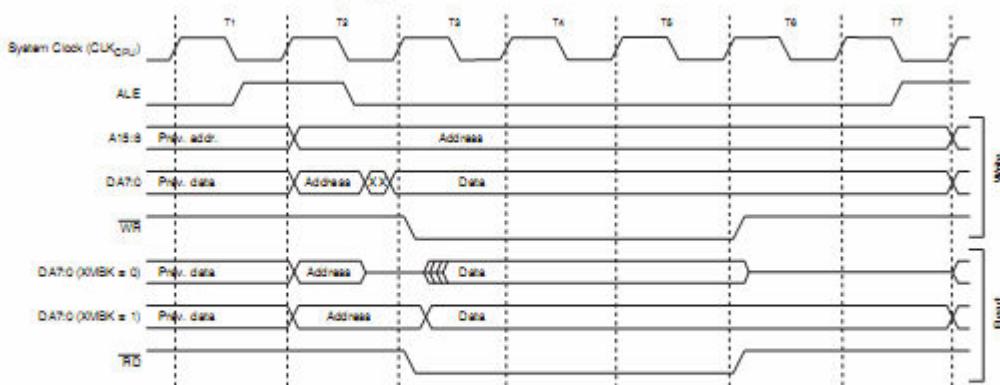
The ALE pulse in period T5 is only present if the next instruction accesses the RAM (internal or external).

**Figure 15.** External Data Memory Cycles with SRWn1 = 1 and SRWn0 = 0<sup>(1)</sup>

Note:

1. SRWn1 = SRW11 (upper sector) or SRW01 (lower sector), SRWn0 = SRW10 (upper sector) or SRW00 (lower sector).

The ALE pulse in period T6 is only present if the next instruction accesses the RAM (internal or external).

**Figure 16.** External Data Memory Cycles with SRWn1 = 1 and SRWn0 = 1<sup>(1)</sup>

Note: 1. SRWn1 = SRW11 (upper sector) or SRW01 (lower sector), SRWn0 = SRW10 (upper sector) or SRW00 (lower sector).  
The ALE pulse in period T7 is only present if the next instruction accesses the RAM (internal or external).

## Miêu tả thanh ghi XMEM (XMEM register Description )

## Thanh ghi điều khiển MCU –MCUCR ( MCU Control Register )

Bit	7	6	5	4	3	2	1	0	MCUCR
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – SRE : kích hoạt SRAM/XMEM bên ngoài

Bằng việc viết SRE là 1 kích hoạt giao diện bộ nhớ ngoài . Các chân chức năng AD7:0 , A15:8 , ALE , WR , và RD được kích hoạt như là các chân chức năng luân phiên . Bít SRE ghi đè lên bất cứ sự cài đặt chân định hướng trong các thanh ghi định hướng dữ liệu tương ứng. Việc viết SRE là 0 , vô hiệu hóa giao diện bộ nhớ ngoài và các chân thông thường và sự cài đặt định hướng dữ liệu được sử dụng .

Bit 6 – SRW10 : bit lựa chọn trạng thái chờ

Để có một miêu tả chi tiết không phải trong chế độ tương thích với Atmega 103 , xem phần mô tả chung cho các bit SRWn bên dưới (XMCRA miêu tả ) . Trong chế độ tương thích với Atmega 103 , việc viết SRW10 là 1 để kích hoạt trạng thái chờ và một chu kỳ xung nhịp được thêm vào trong suốt quá trình phân tích đọc /viết như là được chỉ ra trong hình 14

## Thanh ghi điều khiển bộ nhớ ngoài A – XMCRA

Bit	7	6	5	4	3	2	1	0	XMCRA
Read/Write	-	SRL2	SRL1	SRL0	SRW01	SRW00	SRW11	-	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – RES : bit dự trữ

Đây là một bit dự trữ và sẽ luôn luôn được đọc là 0 . Khi việc viết vào vị trí địa chỉ này , viết bit này là 0 để tương thích với các thiết bị trong tương lai

Bít 6..4 – SRL2 , SRL1 , SRL0 : giới hạn vùng trạng thái chờ

Nó có thể dùng để cấu hình các trạng thái chờ khác nhau cho các địa chỉ bộ nhớ bên ngoài . Không gian địa chỉ bộ nhớ bên ngoài có thể được phân chia vào trong 2 khu vực cái mà vừa phân chia các bit trạng thái chờ . Các bit SRL2 , SRL1 , và SRL0 được đặt là 0 và không gian địa chỉ bộ nhớ bên ngoài nguyên được sử lý như là một khu vực . Khi toàn bộ không gian địa chỉ của SRAM được cấu hình như là một khu , các trạng thái chờ được cấu hình bằng các bit SRW11 và SRW10.

Table 3. Sector limits with different settings of SRL2..0

SRL2	SRL1	SRL0	Sector Limits
0	0	0	Lower sector = N/A Upper sector = 0x1100 - 0xFFFF
0	0	1	Lower sector = 0x1100 - 0x1FFF Upper sector = 0x2000 - 0xFFFF
0	1	0	Lower sector = 0x1100 - 0x3FFF Upper sector = 0x4000 - 0xFFFF
0	1	1	Lower sector = 0x1100 - 0x5FFF Upper sector = 0x6000 - 0xFFFF
1	0	0	Lower sector = 0x1100 - 0x7FFF Upper sector = 0x8000 - 0xFFFF
1	0	1	Lower sector = 0x1100 - 0x9FFF Upper sector = 0xA000 - 0xFFFF
1	1	0	Lower sector = 0x1100 - 0xBFFF Upper sector = 0xC000 - 0xFFFF
1	1	1	Lower sector = 0x1100 - 0xDFFF Upper sector = 0xE000 - 0xFFFF

Bit 1 và bit 6 MCUCR – SRW11 , SRW10 : các bit lựa chọn trạng thái chờ cho các khu vực cao

Các bit điều khiển SRW11 và SRW10 điều khiển thứ tự của các trạng thái chờ cho các khu vực cao hơn trong không gian địa chỉ bộ nhớ ngoài , xem bảng 4 .

Bit 3..2 – SRW01 , SRW00 : bit lựa chọn các trạng thái chờ cho các khu vực thấp hơn

Các bit SRW01 và SRW00 điều khiển thứ tự của các trạng thái chờ cho các khu vực thấp của không gian địa chỉ bộ nhớ ngoài , xem bảng 4

<b>SRWn1</b>	<b>SRWn0</b>	<b>Wait States</b>
0	0	No wait-states
0	1	Wait one cycle during read/write strobe
1	0	Wait two cycles during read/write strobe
1	1	Wait two cycles during read/write and wait one cycle before driving out new address

Note: 1. n = 0 or 1 (lower/upper sector).

For further details of the timing and wait-states of the External Memory Interface, see Figures 13 through Figures 16 for how the setting of the SRW bits affects the timing.

#### Bít 0 – Res : bit dự trữ

Đây là các bit dự trữ và sẽ luôn luôn được đọc là 0 . Khi đọc các vị trí địa chỉ này, viết các bit này là 0 để tương thích với các thiết bị trong tương lai

#### Thanh ghi điều khiển bộ nhớ bên ngoài B - XMCRB

Bit	7	6	5	4	3	2	1	0	XMCRB
Read/Write	XMBK	-	-	-	-	XMM2	XMM1	XMM0	
Initial Value	0	0	0	0	0	0	0	0	

#### Bít 7 – XMBK : kích hoạt bus- keeper bộ nhớ bên ngoài

Việc viết XMBK là 1 để kích hoạt các Bus-keeper trên đường AD7:0 . Khi các Bus –keeper được kích hoạt , nó sẽ đảm bảo 1 mức logic xác định ( 0 hoặc 1 )trên AD7:0 khi chung sẽ khác 3 trạng thái . Việc viết XMBK là 0 để vô hiệu hóa Bus keeper . XMBK thì không đủ điều kiện với SRE , vì vậy dù là giao diện XMEM đã bị vô hiệu hóa , bus keeper thì vẫn hoạt động chỉ cần bit XMBK là 1

#### Bít 6..4 – Res : các bit dự trữ

Đây là các bit dự trữ và sẽ luôn luôn được đọc là 0 . Khi đọc các vị trí địa chỉ này, viết các bit này là 0 để tương thích với các thiết bị trong tương lai

#### Bit 2..0 – XMM2 , XMM1, XMM0 : bộ chẵn mức cao bộ nhớ ngoài

Khi bộ nhớ ngoài được kích hoạt , tất cả các chân cổng C thì được mặc định sử dụng cho các byte địa chỉ cao . Nếu như không gian địa chỉ 60KB đầy thì không cần thiết để truy cập vào các bộ nhớ ngoài , một vài hoặc tất cả , các chân cổng C có thể được giải phóng cho chức năng các chân cổng thông thường như là được mô tả trong bảng 5. Như đã được miêu tả trong phần “sử dụng tất cả 64KB vị trí của bộ nhớ ngoài “ ở trang 35 , nó có khả năng để sử dụng các bit XMMn để truy cập tất cả 64KB vị trí của bộ nhớ ngoài

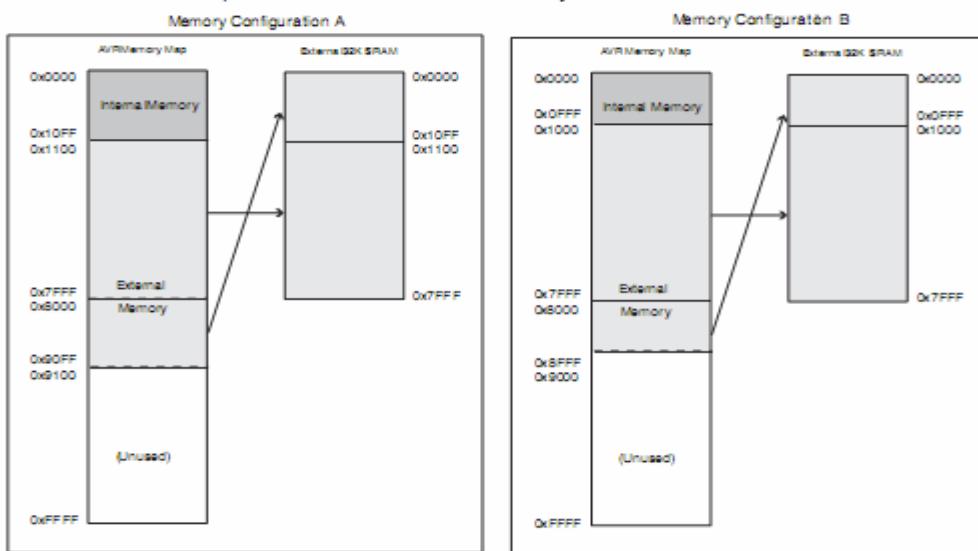
**Table 5.** Port C Pins Released as Normal Port Pins when the External Memory is Enabled

XMM2	XMM1	XMM0	# Bits for External Memory Address	Released Port Pins
0	0	0	8 (Full 60 KB space)	None
0	0	1	7	PC7
0	1	0	6	PC7 - PC6
0	1	1	5	PC7 - PC5
1	0	0	4	PC7 - PC4
1	0	1	3	PC7 - PC3
1	1	0	2	PC7 - PC2
1	1	1	No Address high bits	Full Port C

### Việc sử dụng tất cả các vị trí của bộ nhớ ngoài nhỏ hơn 64KB

Kể từ khi bộ nhớ ngoài được vẽ bản đồ sau bộ nhớ trong như là được chỉ ra trong bảng 11, bộ nhớ ngoài thì không được đánh địa chỉ khi việc đánh địa chỉ 4352 bytes đầu tiên của không gian dữ liệu. Có thể xuất hiện 4352 bytes địa chỉ đầu tiên của bộ nhớ ngoài thì không được truy cập (các địa chỉ của bộ nhớ ngoài là 0x0000 đến 0x10FF). Tuy nhiên, khi kết nối với 1 bộ nhớ ngoài nhỏ hơn 64KB, ví dụ như 32 KB, các vị trí này thì dễ dàng được truy cập 1 cách đơn giản bằng việc đánh địa chỉ từ các địa chỉ 0x8000 đến 0x90FF. Vì rằng các bit địa chỉ A15 của bộ nhớ ngoài thì không được kết nối đến các bộ nhớ ngoài, các địa chỉ từ 0x0000 đến 0x90FF sẽ xuất hiện như là các địa chỉ 0x0000 đến 0x10FF của bộ nhớ ngoài. Việc đánh các địa chỉ bên trên địa chỉ 0x90FF thì không được khuyến nghị, vì rằng điều này sẽ xuất hiện 1 vị trí bộ nhớ ngoài cái mà sẵn sàng được truy cập bởi các địa chỉ (thấp hơn) khác. Đến các phần mềm ứng dụng, 32 KB bộ nhớ bên ngoài sẽ xuất hiện như là 1 không gian địa chỉ 32 KB từ 0x1100 đến 0x90FF. Điều này được minh họa trong hình 17. Cấu hình bộ nhớ B tham khảo trong chế độ tương thích với Atmega 103, cấu hình A thì không có chế độ tương thích.

Khi mà thiết bị được cài đặt trong chế độ tương thích với Atmega 103, các không gian địa chỉ bên trong là 4096 bytes. Điều này kéo theo 4096 bytes địa chỉ đầu tiên của bộ nhớ ngoài có thể bị được truy cập các địa chỉ 0x8000 đến 0x8FFF. Về phần Các phần mềm ứng dụng, bộ nhớ ngoài 32 KB sẽ xuất hiện như là 1 không gian địa chỉ từ 0x1000 đến 0x8FFF

**Figure 17. Address Map with 32 KB External Memory**

### Sử dụng tất cả 64KB vị trí của bộ nhớ ngoài

Vì rằng Bộ nhớ ngoài được vẽ bản đồ sau bộ nhớ trong như là được chỉ ra trong hình 11 , chỉ có 60KB của bộ nhớ ngoài thì được sử dụng 1 cách mặc định (không gian địa chỉ 0x0000 đến 0x10FF được dự trù cho bộ nhớ trong ) Tuy nhiên nó có thể tạo ưu thế cho toàn bộ bộ nhớ ngoài bằng việc giấu các bit địa chỉ cao là 0 . Điều này có thể thực hiện bằng cách sử dụng các bit XMMn và điều khiển bằng phần mềm các bit có giá trị cao nhất của vùng địa chỉ . Bằng việc cài đặt cổng C để xuất ra 0x00 , và sự giải phóng các bit có giá trị cao nhất cho quá trình điều khiển chân cổng bình thường , giao diện bộ nhớ sẽ là địa chỉ 0x0000 đến 0x1FFF . xem đoạn mã mẫu bên dưới để biết thêm chi tiết

**Assembly Code Example<sup>(1)</sup>**

```

; OFFSET is defined to 0x2000 to ensure
; external memory access
; Configure Port C (address high byte) to
; output 0x00 when the pins are released
; for normal Port Pin operation

ldi r16, 0xFF
out DDRC, r16
ldi r16, 0x00
out PORTC, r16
; release PC7:5
ldi r16, (1<<XMM1) | (1<<XMM0)
sts XMCRB, r16
; write 0xAA to address 0x0001 of external
; memory
ldi r16, 0xaa
sts 0x0001+OFFSET, r16
; re-enable PC7:5 for external memory
ldi r16, (0<<XMM1) | (0<<XMM0)
sts XMCRB, r16
; store 0x55 to address (OFFSET + 1) of
; external memory
ldi r16, 0x55
sts 0x0001+OFFSET, r16

```

**C Code Example<sup>(1)</sup>**

```

#define OFFSET 0x2000

void XRAM_example(void)
{
    unsigned char *p = (unsigned char *) (OFFSET + 1);

    DDRC = 0xFF;
    PORTC = 0x00;

    XMCRB = (1<<XMM1) | (1<<XMM0);

    *p = 0xaa;

    XMCRB = 0x00;

    *p = 0x55;
}

```

Note: 1. See "About Code Examples" on page 9.

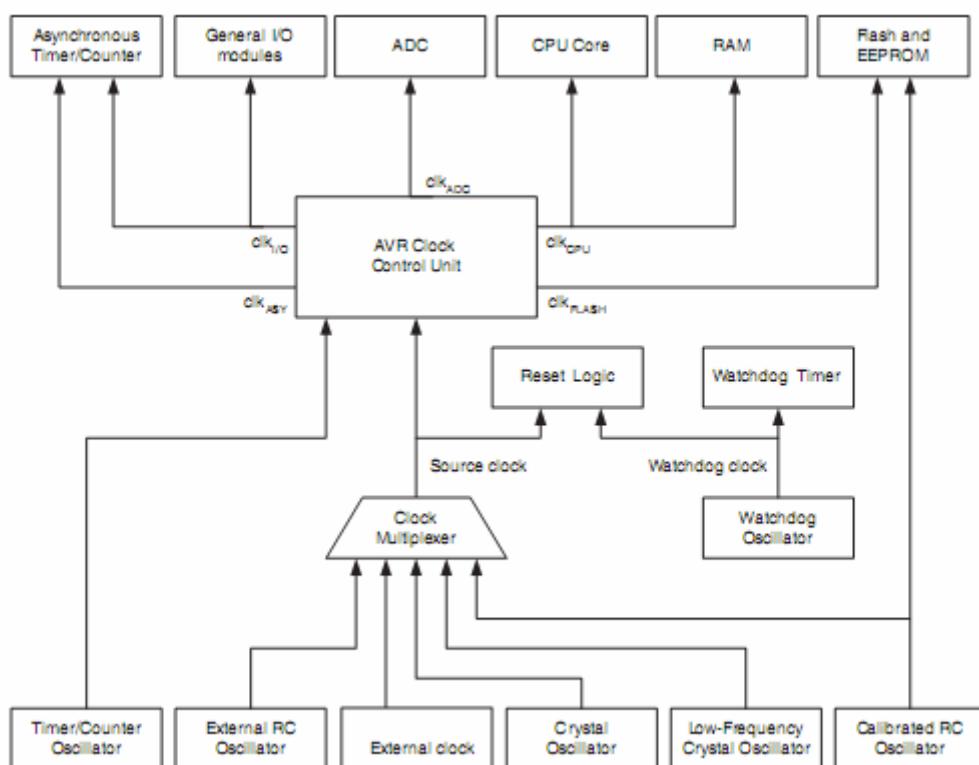
Phải thực hành một cách cẩn thận việc sử dụng lựa chọn này như hầu hết các bộ nhớ được che chắn theo các cách khác nhau .

## V. Xung nhịp hệ thống và lựa chọn xung nhịp

### Xung nhịp hệ thống và sự phân bố của chúng

Hình 18 giới thiệu các xung nhịp hệ thống cơ bản trong AVR và sự phân bố của chúng . Tất cả các xung nhịp đều cần được kích hoạt tại một thời điểm đã được đưa ra . Để giảm sự suy giảm điện áp nguồn , bộ định thời và các modules đang không được sử dụng có thể được dừng bằng cách sử dụng các chế độ ngủ khác nhau như là được mô tả trong phần ‘ sự quản lí nguồn và các chế độ ngủ “ ở trang 45 bộ tạo xung nhịp hệ thống được mô tả chi tiết như dưới đây :

**Figure 18. Clock Distribution**



### Xung nhịp CPU – $\text{clk}_{\text{CPU}}$

Xung nhịp CPU thì được định vị tới 1 phần của hệ thống được hòa hợp với quá trình điều khiển của lõi AVR . Ví dụ của những module này là các file thanh ghi đa năng dùng chung , thanh ghi trạng thái và bộ nhớ dữ liệu đang giữ trong con trỏ ngăn xếp . Việc dừng xung nhịp CPU sẽ ngăn cản lõi khỏi quá trình tính toán và điều khiển chung đang được tiến hành .

## Xung nhịp I/O – $\text{clk}_{\text{I/O}}$

Xung nhịp I/O được sử dụng bởi đa số các module I/O , giống như Timer/Counter , SPI và USART . Xung nhịp I/O cũng được sử dụng bởi các module ngắt ngoài , nhưng chú ý rằng 1 vài ngắt ngoài được dò bởi các logic dị bộ , sự cho phép các ngắt như vậy được tìm kiếm cho dù các xung nhịp I/O bị tạm dừng . Cũng chú ý rằng các sự nhận biết các địa chỉ ở trong module TWI được tiến hành một cách không đồng bộ khi mà  $\text{clk}_{\text{I/O}}$  bị tạm dừng , đang kích hoạt sự thu nhận địa chỉ TWI ở trong tất cả các chế độ sleep .

## Xung nhịp Flash – $\text{clk}_{\text{FLASH}}$

Quá trình điều khiển xung nhịp Flash của giao diện Flash . Xung nhịp Flash thì thường hoạt động cùng lúc với xung nhịp CPU .

## Xung nhịp của Timer dị bộ - $\text{clk}_{\text{ASY}}$

Xung nhịp của timer dị bộ cho phép các bộ timer/counter dị bộ được giữ nhịp 1 cách trực tiếp từ một bộ tạo xung nhịp thạch anh 32 kHz bên ngoài. Vùng xung nhịp xác định này cho phép việc sử dụng Timer/counter này như là 1 bộ đếm thời gian thực ngay cả khi thiết bị này ở trong chế độ ngủ .

## Xung nhịp ADC – $\text{clk}_{\text{ADC}}$

ADC được cung cấp với 1 vùng xung nhịp riêng . Điều này cho phép quá trình dừng CPU và các xung nhịp I/O để mà làm giảm nhiễu sinh ra bởi mạch điện kỹ thuật số . Nó đưa ra được nhiều kết quả đúng đắn hơn trong quá trình chuyển đổi ADC

## Thanh ghi điều khiển chia XTAL – XDIV

Thanh ghi điều khiển phân chia XTAL được sử dụng để phân chia tần số xung nhịp của nguồn bằng 1 số trong khoảng 2-129 . Tính năng này có thể được sử dụng để làm giảm sự sụt áp khi mà có sự cần thiết cho nguồn đang tính toán thấp .

Bit	7	6	5	4	3	2	1	0	XDIV
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

### Bít 7 – XDIVEN : Kích hoạt phân chia XTAL

Khi bit XDIVEN được viết là 1 , tần số xung nhịp của CPU và tất cả các ngoại vi (  $\text{clk}_{\text{I/O}}$  ,  $\text{clk}_{\text{ADC}}$  ,  $\text{clk}_{\text{CPU}}$  ,  $\text{clk}_{\text{FLASH}}$  ) được phân chia bằng việc xác định các hệ số bằng việc cài đặt của XDIV6 – XDIV0 . Bit này có thể được viết trong thời gian thi hành để làm thay đổi tần số xung nhịp để thích hợp với ứng dụng .

### Bít 6..0 – XDIV0 : Bit 6..0 lựa chọn chia XTAL

Các bit này xác định hệ số của sự chia cái mà được áp dụng khi bit XDIVEN được đặt là 1 . Nếu giá trị của các bít được kí hiệu là d , công thức dưới đây xác định kết quả tần số xung nhịp của CPU và các ngoại vi f<sub>CLK</sub> :

$$f_{\text{CLK}} = \frac{\text{xung nhịp nguồn}}{129 - d}$$

Giá trị của các bit này có thể chỉ bị thay đổi khi mà bit XDIVEN được đặt là 0 . Khi mà XDIVEN được viết là 1 , giá trị này được viết đồng thời vào trong XDIV6 ...XDIV0 được chia theo hệ số chia . Khi mà XDIVEN được ghi là 0 , giá trị này được viết đồng thời vào XDIV6..XDIV0 được loại bỏ . Như vậy bộ chia đã chia xung đầu vào chính đến MCU , tốc độ của tất cả các ngoại vi được làm giảm xuống khi mà 1 quá trình chia theo hệ số được sử dụng

Khi xung nhịp hệ thống bị chia , timer/counter 0 có thể được chỉ sử dụng với xung nhịp không đồng bộ này . Tần số của xung nhịp dị bộ này phải thấp hơn 1/4 của tần số của xung nhịp nguồn khi (scaled down ) . Theo cách khác , các ngắt có thể bị mất , và sự truy cập vào thanh ghi timer/counter 0 có thể bị hỏng .

### Xung nhịp nguồn

Thiết bị này có các lựa chọn xung nhịp nguồn dưới đây , có thể lựa chọn bằng bít cầu chì Flash (Flash Fuse ) như là được chỉ ra ở dưới đây . Xung nhịp từ nguồn được lựa chọn nhập vào bộ phát xung nhịp của AVR , được định hướng đến các module tương thích .

**Table 6. Device Clocking Options Select**

Device Clocking Option	CKSEL3..0 <sup>(1)</sup>
External Crystal/Ceramic Resonator	1111 - 1010
External Low-frequency Crystal	1001

**Table 6. Device Clocking Options Select**

Device Clocking Option	CKSEL3..0 <sup>(1)</sup>
External RC Oscillator	1000 - 0101
Calibrated Internal RC Oscillator	0100 - 0001
External Clock	0000

Note: 1. For all fuses "1" means unprogrammed while "0" means programmed.

Có nhiều sự lựa chọn khác nhau cho mỗi sự lựa chọn xung nhịp khác nhau được đưa ra ở phần dưới đây . Khi mà CPU thức dậy từ chế độ ngắt điện hoặc chế độ tiết kiệm điện , nguồn xung nhịp được lựa chọn thì được sử dụng cho thời gian khởi động , bảo đảm sự hoạt động ổn định của bộ tạo dao động trước khi lệnh được bắt đầu . Khi mà CPU khởi động từ chế độ Reset cho phép 1 khoảng thời gian trễ thêm vào để nguồn tiến đến cấp ổn định trước khi bắt đầu quá trình hoạt động bình thường . Bộ tạo dao động Watchdog được dùng để định thời cho phần thời gian thực của thời gian khởi động . Số

lượng của các xung nhịp của bộ dao động WDT được sử dụng cho mỗi khoảng thời gian chờ được chỉ ra trong bảng 7 . Tần số của bộ tạo dao động watchdog là điện áp phụ thuộc như là được chỉ ra trong phần “các chỉ số thông dụng “ ở trang 333

**Table 7. Number of Watchdog Oscillator Cycles**

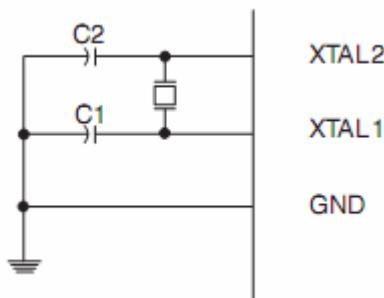
Typical Time-out ( $V_{CC} = 5.0V$ )	Typical Time-Out ( $V_{CC} = 3.0V$ )	Number of Cycles
4.1 ms	4.3 ms	4K (4,096)
65 ms	69 ms	64K (65,536)

Nguồn tạo xung mặc định vận chuyển bởi CKSEL = “0001” và SUT = 10. Sự cài đặt nguồn tạo xung là mặc định vì vậy bộ tạo dao động RC bên trong với khoảng thời gian khởi động là dài nhất . Sự cài đặt mặc định này bảo đảm rằng người sử dụng có thể tạo ra nguồn tạo xung nhịp theo mong muốn của họ bằng cách sử dụng một ứng dụng trong hệ thống (in-system ) hoặc là một phần mềm lập trình song song .

## Bộ tạo dao động thạch anh

XTAL1 và XTAL2 là đầu vào và đầu ra , tương thích , của một bộ khuỷch đại ngược cái mà có thể được cấu hình để sử dụng như là 1 bộ tạo dao động trên chip , như là được chỉ ra trong hình 19 . Một bản tinh thể thạch anh hoặc 1 bộ cộng hưởng gồm có thể được sử dụng . Cầu chì CKOPT lựa chọn giữa 2 kiểu khuỷch đại tạo dao động khác nhau . Khi mà CKOPT được lập trình , đầu ra của bộ tạo dao động sẽ dao động và 1 dao động rail-to-rail ở đầu ra . Kiểu này thì thích hợp khi điều khiển trong 1 môi trường nhiều nhiễu hoặc khi mà đầu ra từ XTAL2 điều khiển 1 bộ đếm xung nhịp thứ 2 . Chế độ này có một khoảng tần số rộng. Khi mà CKOPT không được lập trình bộ tạo dao động có một biên độ dao động đầu ra nhỏ . Điều này làm giảm mức tiêu thụ của nguồn có thể được xét đến . Chế độ này có một dải tần số giới hạn và nó không thể được sử dụng để điều khiển các bộ đếm xung nhịp khác .

Về bộ cộng hưởng , tần số tối đa là 8 MHz với CKOPT không được lập trình và 16 MHz với CKOPT đã được lập trình . C1 và C2 nên luôn luôn được tính toán cho cả hai ,bộ tạo dao động thạch anh và bộ cộng hưởng . Giá trị tốt nhất của tụ điện phụ thuộc vào bộ tạo dao động thạch anh và bộ cộng hưởng được sử dụng , độ lớn của điện dung , và nhiễu điện từ của môi trường . Một vài đường dẫn ban đầu cho sự lựa chọn tụ điện để sử dụng với các bộ tạo dao động thạch anh được đưa ra trong bảng 8 . Về bộ cộng hưởng gồm , các giá trị của tụ được đưa ra bởi nhà sản xuất nên được sử dụng.



các bộ tạo dao động có thể được điều khiển trong 3 chế độ khác nhau , mỗi chế độ thì tối ưu cho một dải tần số xác định . Các chế độ điều khiển thì được lựa chọn bằng cầu chì CKSEL3..1 như là được chỉ ra trong bảng 8

CKOPT	CKSEL3..1	Frequency Range (MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals
1	101 <sup>(1)</sup>	0.4 - 0.9	-
1	110	0.9 - 3.0	12 pF - 22 pF
1	111	3.0 - 8.0	12 pF - 22 pF
0	101, 110, 111	1.0 -	12 pF - 22 pF

Note: 1. This option should not be used with crystals, only with ceramic resonators.

The CKSEL0 fuse together with the SUT1..0 fuses select the start-up times as shown in Table 9.

Bảng 9

Table 9. Start-up Times for the Crystal Oscillator Clock Selection

CKSEL0	SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset ( $V_{cc} = 5.0V$ )	Recommended Usage
0	00	258 CK <sup>(1)</sup>	4.1 ms	Ceramic resonator, fast rising power
0	01	258 CK <sup>(1)</sup>	65 ms	Ceramic resonator, slowly rising power
0	10	1K CK <sup>(2)</sup>	-	Ceramic resonator, BOD enabled
0	11	1K CK <sup>(2)</sup>	4.1 ms	Ceramic resonator, fast rising power
1	00	1K CK <sup>(2)</sup>	65 ms	Ceramic resonator, slowly rising power
1	01	16K CK	-	Crystal Oscillator, BOD enabled
1	10	16K CK	4.1 ms	Crystal Oscillator, fast rising power
1	11	16K CK	65 ms	Crystal Oscillator, slowly rising power

Notes: 1. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.  
 2. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

## Bộ tạo dao động thạch anh tần số thấp

Để sử dụng 1 đồng hồ thạch anh 32768 kHz như là bộ tạo xung nhịp cho thiết bị thì bộ tạo dao động thạch anh tần số thấp phải được lựa chọn bằng việc cài đặt các cầu chì CKSEL đến “1001”. Bộ tạo dao động thạch anh nên được kết nối như hình 19 . Bằng việc lập trình cầu chì CKOPT , người sử dụng có thể kích hoạt các tụ bên trong trên XTAL1 và XTAL2 , như vậy sự gỡ bỏ các tụ bên ngoài là cần thiết . Các tụ bên trong có một giá trị danh định là 36pF.

Khi bộ tạo dao động này được lựa chọn , thời gian khởi động thì được xác định rõ bằng cầu chì SUT như bảng 10

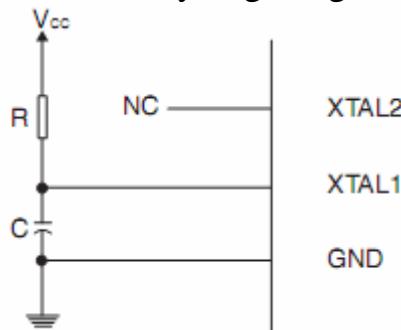
**Table 10.** Start-up Times for the Low-frequency Crystal Oscillator Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset ( $V_{cc} = 5.0V$ )	Recommended Usage
00	1K CK <sup>(1)</sup>	4.1 ms	Fast rising power or BOD enabled
01	1K CK <sup>(1)</sup>	65 ms	Slowly rising power
10	32K CK	65 ms	Stable frequency at start-up
11	Reserved		

Note: 1. These options should only be used if frequency stability at start-up is not important for the application.

## Bộ tạo dao động RC bên ngoài

Để định thời các ứng dụng không nhạy, bộ tạo dao động RC được cấu hình như hình 20 có thể được sử dụng. Tần số được ước lượng một cách đại khái bằng công thức  $f=1/(3RC)$ . C nên để trên 22pF. Bằng việc lập trình cầu chì CKOPT, người sử dụng có thể kích hoạt tụ 36pF bên trong giữa XTAL1 và GND. Vì vậy việc gỡ bỏ bộ tụ bên ngoài là cần thiết. Để biết thông tin về quá trình điều khiển bộ tạo dao động và chi tiết cách chọn R, C, tham khảo chú ý ứng dụng bộ tạo dao động RC bên ngoài.



Bộ tạo dao động có thể hoạt động trong 4 chế độ khác nhau, mỗi 1 chế độ thì tối ưu cho 1 khoảng tần số xác định. Chế độ điều khiển thì được lựa chọn bằng cầu chì CKSEL3..0 trong bảng 11

**Table 11.** External RC Oscillator Operating Modes

CKSEL3..0	Frequency Range (MHz)
0101	0.1 - 0.9
0110	0.9 - 3.0
0111	3.0 - 8.0
1000	8.0 - 12.0

Khi bộ tạo dao động này được lựa chọn, thời gian khởi động thì được xác định rõ bằng cầu chì SUT như bảng 12

**Table 12. Start-Up Times for the External RC Oscillator Clock Selection**

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset ( $V_{CC} = 5.0V$ )	Recommended Usage
00	18 CK	–	BOD enabled
01	18 CK	4.1 ms	Fast rising power
10	18 CK	65 ms	Slowly rising power
11	6 CK <sup>(1)</sup>	4.1 ms	Fast rising power or BOD enabled

Note: 1. This option should not be used when operating close to the maximum frequency of the device.

## Bộ tạo dao động bên trong đã hiệu chỉnh

Bộ tạo dao động bên trong đã hiệu chỉnh cung cấp 1 xung nhịp ổn định 1.0, 2.0, 4.0, 8.0 MHz . Tất cả các tần số là giá trị danh định ở 5V và 25°C . Xung nhịp này có thể được lựa chọn như xung nhịp hệ thống bằng cách lập trình cầu chì CKSEL như trong bảng 13 . Nếu được chọn lựa , nó sẽ hoạt động mà không có bộ phận bên ngoài nào . Cầu chì CKOPT nên luôn luôn không lập trình khi sử dụng lựa chọn xung nhịp này . Trong suốt quá trình Reset , phần cứng tải các byte hiệu chỉnh này cho 1 bộ tạo dao động 1 MHz ở trong thanh ghi OSCCAL và do đó tự động hiệu chỉnh bộ tạo dao động RC . Ở 5V , 25độ và tần số bộ tạo dao động 1.0MHz được lựa chọn , quá trình hiệu chỉnh này đưa ra 1 tần số trong khoảng ± 3% của tần số danh định . Việc sử dụng phương pháp hiệu chỉnh này như được miêu tả trên [vWWW.atmel.com](http://WWW.atmel.com) là có thể đạt được độ chính xác ± 1% ở bất cứ điện áp , và nhiệt độ nào . Khi bộ tạo dao động này được sử dụng như là một xung nhịp của chip , bộ tạo dao động Watchdog sẽ vẫn được sử dụng cho Timer Watchdog và cho thời gian chờ reset ( Reset Time-out ) . Để biết thêm thông tin chi tiết về giá trị hiệu chỉnh tiền lập trình , xem phần “calibration Byte “ ở trang 289

**Table 13. Internal Calibrated RC Oscillator Operating Modes**

CKSEL3..0	Nominal Frequency (MHz)
0001 <sup>(1)</sup>	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. The device is shipped with this option selected.

Khi mà bộ tạo dao động được lựa chọn , thời gian khởi động được xác định rõ bằng cầu SUT như bảng 14 . XTAL1 và XTAL2 nên được cho phép không kết nối .

**Table 14.** Start-up Times for the Internal Calibrated RC Oscillator Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset ( $V_{CC} = 5.0V$ )	Recommended Usage
00	6 CK	–	BOD enabled
01	6 CK	4.1 ms	Fast rising power
10 <sup>(1)</sup>	6 CK	65 ms	Slowly rising power
11	Reserved		

Notes: 1. The device is shipped with this option selected.

## Thanh ghi hiệu chỉnh bộ tạo dao động – OSCCAL

Bit	7	6	5	4	3	2	1	0	OSCCAL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	Device Specific Calibration Value								

Chú ý : thanh ghi OSCCAL thì không khả dụng ở trong chế độ tương thích với Atmega103

Bít 7..0 – CAL7..0 : giá trị điều chỉnh của bộ tạo dao động

Việc viết byte hiệu chỉnh vào địa chỉ này sẽ hoàn thiện bộ tạo dao động trong để gỡ bỏ tiến trình biến thiên khỏi tần số bộ tạo dao động Oscillator . Trong suốt quá trình Reset , 1 giá trị hiệu chỉnh 1 MHz cái mà được định vị trong dòng kí hiệu byte cao ( địa chỉ 0x00 ) thì được tải tự động vào trong thanh ghi OSCCAL . Nếu bộ tạo dao động bên trong được sử dụng tại các tần số khác , giá trị hiệu chỉnh phải được tải 1 cách thủ công bằng tay . Điều này có thể được thực hiện trước tiên bằng việc đọc các dòng tín hiệu bởi 1 lập trình viên , và sau đó lưu các giá trị hiệu chỉnh ở trong bộ nhớ Flash và EEPROM . Sau đó các giá trị này có thể được đọc bằng phần mềm và được tải vào trong thanh ghi OSCCAL . Khi OSCCAL là không , tần số có ích thấp nhất được chọn . Việc viết các giá trị không phải bằng không vào thanh ghi này sẽ làm tăng tần số của bộ tạo dao động bên trong . Việc viết \$FF vào thanh ghi sẽ đưa ra tần số khả dụng cao nhất . Bộ tạo dao động đã được hiệu chỉnh thì được sử dụng trong thời gian truy cập bộ nhớ Flash và EEPROM . Nếu EEPROM và Flash đã được ghi , không được điều chỉnh lớn hơn 10% trên tần số danh định . Nói cách khác , sự ghi EEPROM và Flash có thể bị hỏng . Chú ý rằng bộ tạo dao động nhằm mục đích điều chỉnh 1.0 , 2.0 , 4.0 , hoặc 8.0 MHz . Việc điều chỉnh sang các giá trị khác thì không được bảo đảm , như là được biểu diễn ở bảng 15

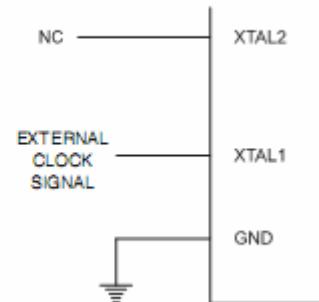
**Table 15.** Internal RC Oscillator Frequency Range.

OSCCAL Value	Min Frequency in Percentage of Nominal Frequency (%)	Max Frequency in Percentage of Nominal Frequency (%)
\$00	50	100
\$7F	75	150
\$FF	100	200

## Xung nhịp bên ngoài

Để điều khiển các thiết bị từ một nguồn tạo xung nhịp bên ngoài , XTAL1 nên được điều khiển như được chỉ ra trong hình 21 . Để chạy thiết bị này trên 1 xung nhịp ngoài , cầu chì CKSEL phải được lập trình là “0000” . Bằng việc lập trình cầu chì CKOPT , người sử dụng có thể kích hoạt 1 bộ tụ 36pF bên trong giữa XTAL1 và GND

**Figure 21.** External Clock Drive Configuration



Khi mà bộ tạo dao động được lựa chọn , thời gian khởi động được xác định rõ bằng cầu SUT như bảng 16 .

**Table 16.** Start-up Times for the External Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset ( $V_{CC} = 5.0V$ )	Recommended Usage
00	6 CK	–	BOD enabled
01	6 CK	4.1 ms	Fast rising power
10	6 CK	65 ms	Slowly rising power
11	Reserved		

Khi áp dụng 1 xung nhịp bên ngoài , cần thiết tránh các thay đổi đột ngột trong quá trình áp dụng tần số xung nhịp để đảm bảo hoạt động ổn định của MCU . Một sự thay đổi tần số lớn hơn 2% từ 1 chu kì xung nhịp đến xung kế tiếp có thể dẫn đến các trạng thái không thể tiên đoán trước . Điều này là cần thiết để đảm bảo rằng MCU được giữ ở chế độ Reset trong suốt quá trình thay đổi tần số xung nhịp .

## Bộ tạo dao động Timer/Counter

Vi điều khiển AVR với các chân của bộ tạo dao động Timer/Counter (TOSC1 và TOSC2) , bộ dao động thạch anh được nối trực tiếp giữa các chân . Không có các tụ ngoài nào là cần thiết . Bộ tạo dao động thì tối ưu cho việc sử dụng với bộ tạo dao động theo dõi thạch anh 32.768kHz . Việc áp dụng 1 nguồn xung nhịp ngoài đến TOSC1 thì không được khuyến cáo .

Chú ý : bộ tạo dao động Timer/Counter sử dụng bộ tạo dao động thạch anh giống như bộ tạo dao động tần số thấp và các tụ bên trong có cùng các giá trị danh định là 36pF

## VI . Quản lý nguồn điện và các chế độ sleep

Chế độ chờ sleep kích hoạt chương trình ứng dụng để tắt các module không sử dụng trong MCU , so vậy tiết kiệm được nguồn điện . AVR cung cấp nhiều chế độ sleep cho phép người dùng điều chỉnh sự tốn hao nguồn điện đáp ứng đòi hỏi của các chương trình ứng dụng .

Để truy nhập vào bất cứ chế độ nào trong số 6 chế độ chờ , bit SE ở trong thanh ghi MCUCR phải được ghi với biến logic là 1 và 1 lệnh SLEEP phải được thực thi . Các bit SM2 , SM1 , và SM0 trong thanh ghi MCUCR lựa chọn chế độ chờ (Idle , giảm nhiễu ADC , tắt nguồn , tiết kiệm điện , chờ , hoặc chế độ chờ mở rộng ) sẽ được kích hoạt bằng lệnh SLEEP . xem bảng 17 để biết thêm chi tiết . Nếu một ngắt kích hoạt xuất hiện trong khi MCU đang ở trong chế độ chờ , MCU được đánh thức. MCU sau đó được tạm dừng trong 4 chu kỳ xung nhịp trong thời gian khởi động được thêm vào , nó thực thi chương trình con phục vụ ngắn , và khôi phục lại quá trình thực thi từ các lệnh kế tiếp lệnh SLEEP . Thành phần của thanh ghi file và SRAM thì không bị thay đổi khi mà thiết bị được đánh thức từ chế độ chờ . Nếu như tín hiệu reset xuất hiện trong suốt chế độ chờ , MCU được đánh thức và được thực thi từ các vecto Reset .

Hình 18 trên trang 36 giới thiệu các xung nhịp hệ thống khác nhau trong Atmega 128 , và sự phân bố của chúng . Hình này cũng rất hữu dụng trong việc lựa chọn 1 chế độ chờ thích hợp.

### Thanh ghi điều khiển MCU

Thanh ghi điều khiển MCU bao gồm các bit điều khiển cho sự quản lý nguồn điện

Bit	7	6	5	4	3	2	1	0	MCUCR
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 5 – SE : kích hoạt chế độ chờ

Bít SE phải được ghi ở mức logic 1 để truy nhập vào MCU trong chế độ Chờ khi mà lệnh SLEEP được thực thi . Để tránh việc truy nhập vào MCU trong chế độ chờ trừ phi nó là 1 bộ lập trình đa năng , khuyến nghị để viết bit kích hoạt chế độ Chờ SE là 1 trước khi quá trình thực hiện lệnh SLEEP và xóa nó ngay lập tức khi thức dậy

Bit 4..2 – SM2..0 : Bit 2 ,1 ,0 lựa chọn chế độ chờ

Các bit này lựa chọn giữa 6 trạng thái khả dụng như là được chỉ ra trong bảng 17

**Table 17. Sleep Mode Select**

<b>SM2</b>	<b>SM1</b>	<b>SM0</b>	<b>Sleep Mode</b>
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby <sup>(1)</sup>
1	1	1	Extended Standby <sup>(1)</sup>

Note: 1. Standby mode and Extended Standby mode are only available with external crystals or resonators.

## Idle mode

Khi mà các bit SM2..0 được viết là 000 , lệnh ngủ làm cho MCU truy nhập vào chế độ Idle , việc dừng CPU nhưng cho phép SPI , USART , bộ so sánh tương tự , chuyển đổi ADC giao diện 2 dây nối tiếp , Timer/Counter , Watchdog , và các ngắt hệ thống tiếp tục hoạt động . Chế độ ngủ này cơ bản dừng  $\text{clk}_{\text{CPU}}$  và  $\text{clk}_{\text{FLASH}}$  trong khi cho phép các xung nhịp khác chạy .

Chế độ Idle kích hoạt MCU thức dậy từ các bộ ngắt khởi động bên ngoài thì tốt như là 1 bộ phận bên trong giống như timer tràn và các ngắt hoàn thành quá trình chuyển đổi USART . Nếu nó thức dậy từ các ngắt của bộ so sánh tương tự Analog thì không cần thiết , bộ so sánh Analog có thể tắt nguồn bằng cách cài đặt các bit ACD ở trong bộ điều khiển so sánh analog và thanh ghi trạng thái – ASCR . Nó sẽ giảm nhu cầu dùng điện trong chế độ Idle . Nếu ADC được kích hoạt , 1 quá trình chuyển đổi bắt đầu tự động khi mà chế độ này được truy nhập.

## Chế độ giảm nhiễu ADC

Khi mà bit SM2..0 được đặt là001 , lệnh SLEEP làm cho MCU truy nhập vào chế độ giảm nhiễu ADC , việc dừng CPU nhưng cho phép ADC ,các ngắt ngoài , bộ theo dõi địa chỉ giao diện 2 dây nối tiếp , Timer/Counter 0 và Watchdog tiếp tục hoạt động (nếu được kích hoạt). Chế độ ngủ này cơ bản dừng  $\text{clk}_{\text{I/O}}$  ,  $\text{clk}_{\text{CPU}}$  và  $\text{clk}_{\text{FLASH}}$  trong khi cho phép các xung nhịp khác chạy .

Điều này cải tạo môi trường nhiễu cho ADC , và làm cho độ chính xác của các phép đo cao hơn . Nếu như bộ chuyển đổi ADC được kích hoạt thì , một sự chuyển đổi bắt đầu tự động khi mà chế độ này được truy nhập . Một phần ngắt bổ sung của bộ chuyển đổi ADC , chỉ có 1Reset ngoài , 1Reset Watchdog , 1 Reset tắt nguồn ,1 địa chỉ giao diện 2 dây nối tiếp tương ứng với các ngắt , 1ngắt Timer/Counter0 1 SPM/EEPROM sẵn sàng ngắt , 1 cấp ngắt bên ngoài trên INT7:4 , hoặc 1 ngắt ngoài trên INT3:0 có thể đánh thức MCU từ chế độ giảm nhiễu ADC

## Chế độ ngắt nguồn

Khi mà các bit SM2..0 được viết là 010, lệnh SLEEP làm cho MCU truy nhập vào chế độ Tắt nguồn Power – down . Trong chế độ này , bộ tạo dao động bên ngoài bị dừng lại , trong khi các ngắt ngoài , đồng hồ địa chỉ giao diện 2 dây nối tiếp , và 1 Watchdog vẫn tiếp tục hoạt động (nếu được kích hoạt ) . Chỉ có 1 Reset ngoài , 1 Reset Watchdog , 1 Reset Brown-out , 1 ngắt tương ứng địa chỉ giao diện 2 dây nối tiếp , 1 cấp ngắt ngoài trên INT7:4 , hoặc 1 ngắt ngoài trên INT3:0 có thể đánh thức MCU. Chế độ ngủ này dừng 1 cách cơ bản tất cả các nguồn phát xung nhịp , chỉ cho phép các module dị bộ hoạt động

Chú ý rằng nếu 1 ngắt khởi động cấp được sử dụng cho việc đánh thức từ chế độ ngắt nguồn Power-down , sự thay đổi cấp phải được giữ trong vài giai đoạn để đánh thức MCU . Tham khảo các ngắt ngoài trong trang90 để biết thêm chi tiết .

Khi sự đánh thức từ chế độ ngắt nguồn Power-down , có 1 sự chậm trễ từ trạng thái đánh thức xuất hiện cho đến khi sự đánh thức trở nên có hiệu lực . Điều này cho phép bộ định thời để khởi động và trở nên ổn định sau khi hoàn thành quá trình dừng lại. Chu kì đánh thức được định nghĩa bởi các cầu chì CKSEL giống nhau mà xác định chu kì Reset Time-out , như được miêu tả trong ”Clock Source” trang 37

## Chế độ tiết kiệm điện Power-save

Khi mà các bit SM2..0 được viết là 011 , lệnh SLEEP làm cho MCU đăng nhập vào chế độ tiết kiệm điện Power save . Chế độ này giống với chế độ ngắt nguồn Power-down , với 1 điểm khác sau :

Nếu Timer/Counter 0 bị khóa 1 cách không đồng bộ , ví dụ Bít AS0 trong thanh ghi ASSR được cài đặt , Timer/Counter0 sẽ chạy trong suốt quá trình ngủ. Thiết bị này có thể được đánh thức từ 1 sự tràn Timer hoặc 1 sự kiện so sánh đầu ra từ Timer/Counter0 nếu 1 bít kích hoạt ngắt Timer/Counter tương ứng được cài đặt trong TIMSK , và các bít kích hoạt ngắt chung trong thanh ghi SREG được cài đặt

Nếu Timer dị bộ không bị khóa dị bộ , chế độ ngắt nguồn Power-down được khuyến cáo thay thế cho chế độ tiết kiệm điện Power – save bởi vì thành phần của các thanh ghi trong Timer dị bộ nên được coi như là không xác định sau khi đánh thức trong chế độ tiết kiệm điện nếu AS0 là 0.

Chế độ ngủ này dừng cơ bản tất cả các bộ phát xung nhịp trừ  $\text{clk}_{\text{ASY}}$  ,sự cho phép hoạt động chỉ trong các module dị bộ , bao gồm các Timer/Counter 0 nếu bị khóa dị bộ .

## Chế độ chờ Standby

Khi mà các bit SM2..0 được đặt là 110 và xung nhịp của các bộ tạo dao động thạch anh và bộ cộng hưởng bên ngoài được lựa chọn , lệnh SLEEP làm cho MCU đăng nhập vào chế độ chờ Stand by . Chế độ này tương tự như chế độ ngắt nguồn Power-

down với ngoại lệ là các bộ phát dao động Oscillator được giữ trong quá trình chạy . Từ chế độ chờ Standby , thiết bị được đánh thức trong 6 chu kì xung nhịp đồng hồ

## Chế độ chờ mở rộng

Khi mà các bit SM2..0 được đặt là 111 và 1 xung nhịp của bộ tạo dao động thạch anh/bộ cộng hưởng đã được lựa chọn , lệnh SLEEP làm cho MCU truy nhập vào chế độ chờ mở rộng. Chế độ này thì giống như chế độ tiết kiệm điện với 1 ngoại lệ là bộ tạo dao động thì được giữ trong khi đang chạy . Từ chế độ chờ mở rộng Extended Standby , các thiết bị này được đánh thức trong 6 chu kì xung nhịp.

**Table 18. Active Clock Domains and Wake Up Sources in the Different Sleep Modes**

Sleep Mode	Active Clock Domains					Oscillators		Wake Up Sources					
	clk <sub>CPU</sub>	clk <sub>FLASH</sub>	clk <sub>I/O</sub>	clk <sub>ADC</sub>	clk <sub>ASY</sub>	Main Clock Source Enabled	Timer Osc Enabled	INT7:0	TWI Address Match	Timer 0	SPM/EEPROM Ready	ADC	Other I/O
Idle			X	X	X	X	X <sup>(2)</sup>	X	X	X	X	X	X
ADC Noise Reduction				X	X	X	X <sup>(2)</sup>	X <sup>(2)</sup>	X	X	X	X	
Power-down								X <sup>(3)</sup>	X				
Power-save					X <sup>(2)</sup>		X <sup>(2)</sup>	X <sup>(3)</sup>	X	X <sup>(2)</sup>			
Standby <sup>(1)</sup>						X		X <sup>(3)</sup>	X				
Extended Standby <sup>(1)</sup>					X <sup>(2)</sup>	X	X <sup>(2)</sup>	X <sup>(3)</sup>	X	X <sup>(2)</sup>			

Notes: 1. External Crystal or resonator selected as clock source

2. If AS0 bit in ASSR is set

3. Only INT3:0 or level interrupt INT7:4

## Sự tối thiểu hóa tổn hao công suất nguồn

Có rất nhiều yếu tố phải xem xét khi có gắng làm cực tiểu tổn hao công suất nguồn trong 1 hệ thống được điều khiển bởi AVR . Nói chung ,các chế độ ngủ Sleep nên được sử dụng tối đa có thể ,và chế độ chờ nên được lựa chọn vì vậy một vài chức năng của thiết bị đang hoạt động . Tất cả các chức năng không cần thiết thì nên bị vô hiệu hóa . Đặc biệt , các module dưới đây có thể cần sự xem xét đặc biệt khi cố gắng để đạt được sự tổn hao nguồn thấp nhất có thể

## Bộ chuyển đổi từ Analog sang Digital

Nếu được kích hoạt , bộ chuyển đổi ADC sẽ được kích hoạt ở trong các chế độ ngủ Sleep . Để tiết kiệm nguồn , bộ ADC nên được vô hiệu hóa trước khi truy nhập vào bất cứ chế độ ngủ Sleep nào . Khi bộ ADC được tắt và bật trở lại , sự chuyển đổi tiếp

theo sẽ là 1 sự chuyển đổi mở rộng . Tham khảo phần bộ chuyển đổi từ tương tự sang số ở trang 230 để biết thêm chi tiết hoạt động của ADC .

## Bộ so sánh tương tự

Khi truy nhập vào chế độ Idle , bộ so sánh tương tự nên được vô hiệu hóa nếu được sử dụng . Khi truy nhập vào chế độ giảm nhiễu ADC , bộ so sánh Analog nên được vô hiệu hóa . Trong các chế độ ngủ khác , bộ so sánh Analog bị vô hiệu hóa 1 cách tự động . Tuy nhiên , nếu bộ so sánh Analog được cài đặt để sử dụng như là một điện thế trong tham chiếu như là đầu vào , bộ so sánh Analog nên được vô hiệu hóa trong tất cả các chế độ ngủ . Nói cách khác , điện áp trong tham chiếu sẽ được kích hoạt , phụ thuộc vào chế độ ngủ . Tham khảo bộ so sánh Analog ở trang 227 để biết thêm chi tiết về cách cấu hình bộ so sánh Analog

## Bộ dò sự yếu nguồn Brown-out

Nếu bộ dò Brown-out là không cần thiết trong ứng dụng , module nên được tắt . Nếu bộ dò Brown-out được kích hoạt bằng cầu chì BODEN , nó sẽ được kích hoạt trong tất cả các chế độ ngủ , và do đó , luôn luôn tiêu hao nguồn . Trong các chế độ ngủ sâu hơn ( deeper sleep ) , điều này sẽ đóng góp đáng kể vào tổng tốn hao công suất trong mạch. Tham khảo thêm phần “Brown-out Detector ” để biết thêm chi tiết về cách cấu hình bộ dò sự yếu nguồn

## Điện áp chuẩn bên trong ( Internal Voltage Reference )

Điện áp chuẩn bên trong sẽ được kích hoạt khi cần bởi Bộ dò sự yếu nguồn, bộ so sánh tương tự , và bộ chuyển đổi ADC . Nếu các module này bị vô hiệu hóa như là đã được mô tả trong phần trên , Điện áp tham khảo (Internal Voltage Reference ) sẽ bị vô hiệu hóa và nó sẽ làm tốn hao nguồn . Khi được bật trở lại , người sử dụng phải cho phép truy xuất để khởi động trước khi đầu ra được sử dụng . Nếu sự truy xuất này bị giữ lại trong chế độ ngủ thì đầu ra có thể được sử dụng ngay lập tức . Tham khảo thêm phần ‘Internal Voltage Reference “ ở trang 54 để biết thêm chi tiết về thời gian khởi động

## Timer Watchdog

Nếu timer watchdog không cần thiết trong chương trình ứng dụng thì module này nên được tắt . Nếu như Timer watchdog được kích hoạt, nó sẽ được kích hoạt trong tất cả các chế độ ngủ , và do đó nó luôn luôn tiêu hao công suất nguồn , . Trong các chế độ ngủ sâu hơn , điều này sẽ góp phần đáng kể vào tổng tốn hao của mạch điện . Tham khảo phần Timer watchdog ở trang 55 để biết thêm chi tiết về cách cấu hình Timer watchdog.

## Các chân cổng

Khi truy nhập vào 1 chế độ ngủ , tất cả các chân cổng nên được cấu hình để tối thiểu hóa công suất sử dụng . Điều quan trọng nhất là sau đó bảo đảm rằng không có chân nào điều khiển các tần thuần trở . Trong các chế độ ngủ khi cả 2 xung nhịp I/O ( $\text{clk}_{\text{I/O}}$ ) và xung nhịp ADC ( $\text{clk}_{\text{ADC}}$ ) bị dừng lại thì bộ đệm đầu vào của thiết bị sẽ bị vô hiệu hóa . Điều này đảm bảo rằng không có công suất nguồn bị tiêu hao bởi đầu vào logic khi không cần thiết. Trong một vài trường hợp, cổng vào logic là cần thiết cho việc dò tìm điều kiện thức dậy (wake-up conditions) và sau đó nó sẽ được kích hoạt lại . Tham khảo thêm phần đầu vào số và các chế độ ngủ ở trang 70 để biết thêm chi tiết về các chân nào được kích hoạt . Nếu bộ đệm đầu ra được kích hoạt và các tín hiệu đầu vào được gỡ bỏ di động (floating) hoặc có 1 cấp tín hiệu tương tự đóng vào chân  $\text{V}_{\text{CC}}/2$  , thì bộ đệm đầu vào sẽ sử dụng nguồn thửa .

## Giao diện JTAG và hệ thống dò lỗi trên chip

Nếu như hệ thống dò lỗi trên chip được kích hoạt bằng cầu chì OCDEN và chip truy nhập vào chế độ ngắt nguồn (Power down) hoặc chế độ ngủ tiết kiệm điện , nguồn phát xung nhịp chính còn lại được kích hoạt . Trong các chế độ ngủ khác , điều này sẽ đóng góp đáng kể vào tổng hao hụt dòng điện . Có 3 cách khác nhau để tránh điều này :

- vô hiệu hóa Cầu chì OCDEN
- Vô hiệu hóa cầu chì JTAGEN
- Viết 1 vào bit JTD ở trong thanh ghi MCUCSR

Chân TDO thì được rời thay đổi khi mà giao diện JTAG được kích hoạt trong khi bộ điều khiển JTAG TAP đang không chuyển dữ liệu . Nếu phần cứng được kết nối với chân TDO không dừng lại ở một mức logic , sự tổn hao công suất nguồn sẽ tăng lên . Chú ý rằng chân TDI cho thiết bị tiếp theo trong 1 chuỗi quét bao gồm 1 sự dừng lại để tránh các vấn đề này . Việc viết bit JTD trong thanh ghi MCUCSR là 1 hoặc di chuyển cầu chì JTAG không được lập trình để vô hiệu hóa giao diện JTAG

## VII . Điều khiển hệ thống và Reset ( System Control and Reset )

### Quá trình Reset của AVR

Trong suốt quá trình Reset , tất cả các thanh ghi I/O được cài đặt đến các giá trị khởi đầu của chúng , và chương trình bắt đầu thực thi từ các vecto Reset của chúng . Các lệnh được đặt ở trong Vecto Reset phải là một JMP – bước nhảy hoàn toàn – chỉ dẫn đến các chương trình con điều khiển quá trình Reset . Nếu như chương trình không bao giờ kích hoạt một nguồn ngắn , véc tơ ngắn không được sử dụng , và đoạn mã chương trình bình thường có thể được đặt tại các vị trí khác nhau . Đây cũng là trường hợp nếu như vecto Reset đang ở trong khu vực ứng dụng trong khi các vecto ngắn đang ở trong khu vực khởi động (Boot Section) hoặc vice versa . Mạch điện ở hình 22 chỉ ra mức logic Reset . bảng 19 xác định các tham số điện của mạch reset

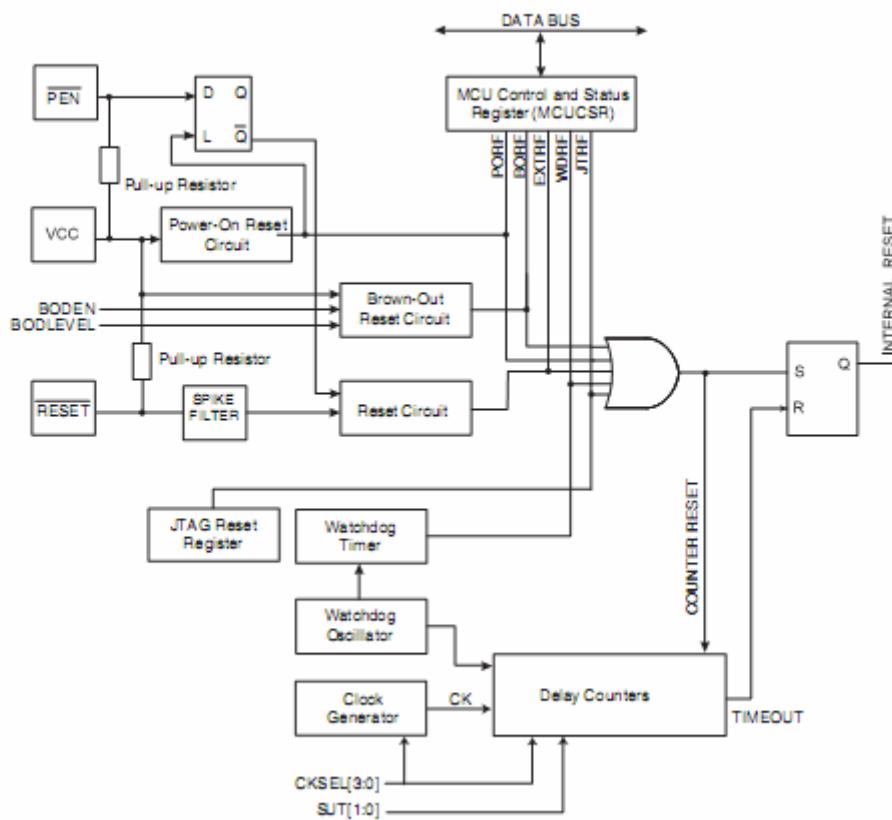
Các cổng I/O của AVR thì reset ngay lập tức về trạng thái ban đầu của chúng khi mà nguồn reset tiến hành hoạt động . Không cần thiết bắt cứ nguồn phát xung nào để chạy

Sau khi nguồn Reset vừa dừng hoạt động , một bộ đếm trễ được gọi ra , kéo dài Reset bên trong . Điều này cho phép nguồn hướng tới cấp độ ổn định trước khi hoạt động được bắt đầu . Chu kỳ định giờ của bộ đếm trễ được xác định bằng người sử dụng thông qua các cầu chì CKSEL . Các lựa chọn khác cho các chu kỳ trễ được giới thiệu trong “clock Source” ở trang 37

### Các nguồn Reset

Atmega 128 có 5 nguồn Reset :

- Reset bật nguồn (Power On Reset). MCU thì Reset khi mà điện áp nguồn cấp thấp hơn ngưỡng Reset bật nguồn (Power On Reset)
- Reset ngoài . MCU thì Reset khi 1 mức thấp được đưa ra trên chân RESET dài hơn độ dài xung tối thiểu
- Reset Watchdog . MCU thì Reset khi mà chu kỳ Timer watchdog hết hạn (expires ) và watchdog được kích hoạt
- Reset Brown-out .MCU sẽ reset khi điện áp nguồn cấp  $V_{CC}$  ở dưới ngưỡng Reset yếu điện áp Brown-out ( $V_{BOT}$ ) và bộ dò yếu điện áp (Brown-out Detector ) được kích hoạt
- Reset JTAG AVR . MCU thì Reset chỉ cần có 1 mức logic 1 ở trên thanh ghi Reset , 1 trong chuỗi quét của hệ thống JTAG . Tham khảo phần IEEE 1149.1 (JTAG) Boundary –scan ở trang 252 để thêm chi tiết

**Figure 22.** Reset Logic

**Table 19. Reset Characteristics**

Symbol	Parameter	Condition	Min	Typ	Max	Units
$V_{POT}$	Power-on Reset Threshold Voltage (rising)			1.4	2.3	V
	Power-on Reset Threshold Voltage (falling) <sup>(1)</sup>			1.3	2.3	V
$V_{RST}$	RESET Pin Threshold Voltage		0.2 $V_{CC}$		0.85 $V_{CC}$	V
$t_{RST}$	Pulse width on RESET Pin		1.5			$\mu s$
$V_{BOT}$	Brown-out Reset Threshold Voltage <sup>(2)</sup>	$BODLEVEL = 1$	2.4	2.6	2.9	V
		$BODLEVEL = 0$	3.7	4.0	4.5	
$t_{BOD}$	Minimum low voltage period for Brown-out Detection	$BODLEVEL = 1$		2		$\mu s$
		$BODLEVEL = 0$		2		
$V_{HYST}$	Brown-out Detector hysteresis			100		mV

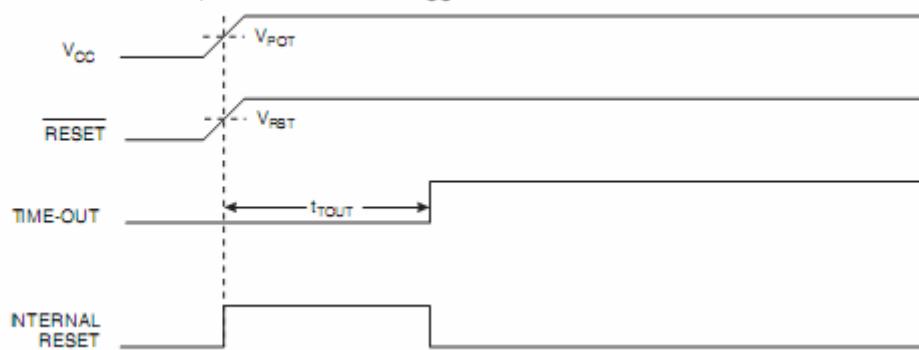
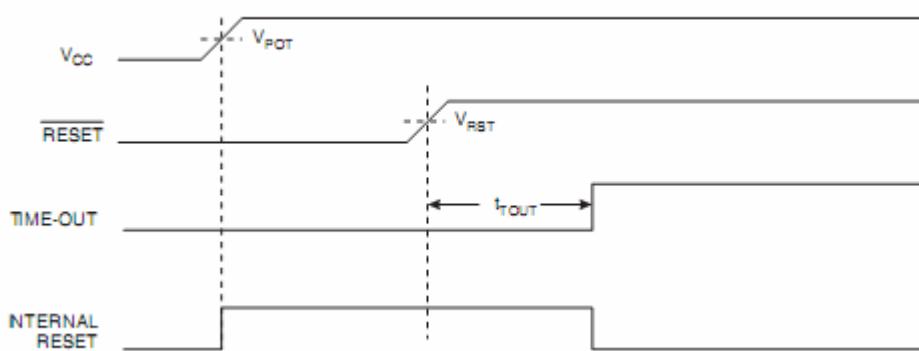
Notes: 1. The Power-on Reset will not work unless the supply voltage has been below  $V_{POT}$  (falling)

2.  $V_{BOT}$  may be below nominal minimum operating voltage for some devices. For devices where this is the case, the device is tested down to  $V_{CC} = V_{BOT}$  during the production test. This guarantees that a Brown-out Reset will occur before  $V_{CC}$  drops to a voltage where correct operation of the microcontroller is no longer guaranteed. The test is performed using  $BODLEVEL=1$  for ATmega128L and  $BODLEVEL=0$  for ATmega128.  $BODLEVEL=1$  is not applicable for ATmega128

## Reset Power on

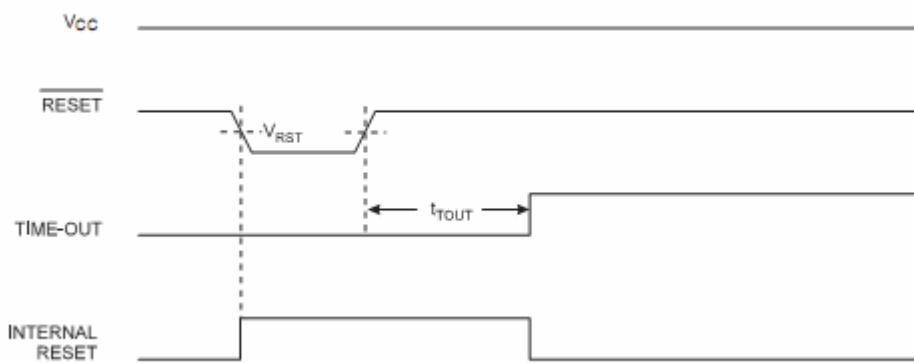
1 xung reset bật nguồn (POR) (Reset Power on) sẽ được phát ra bởi mạch dò trên chip . Cấp độ phát hiện được định nghĩa ở bảng 19 . Xung POR được kích hoạt ở bất cứ đâu  $V_{CC}$  cấp độ phát hiện thấp . Một mạch POR có thể được sử dụng để khởi động Reset Start-up , như là để phát hiện 1 sự hỏng hóc ở điện áp nguồn cấp .

Một mạch Reset bật nguồn (Reset Power on) (POR) bảo đảm rằng các thiết bị được reset từ lúc bật nguồn . Khoảng ngưỡng của điện áp Reset bật nguồn được đưa ra trong bộ đếm trễ , cái mà xác định khoảng thời gian bao lâu để thiết bị được giữ trong chế độ RESET sau khi mà  $V_{CC}$  tăng lên . Tín hiệu Reset được kích hoạt trở lại , mà không có bất cứ trễ nào , khi mà  $V_{CC}$  giảm dưới mức phát hiện (detection level )

**Figure 23.** MCU Start-up, RESET Tied to  $V_{CC}$ .**Figure 24.** MCU Start-up, RESET Extended Externally

## Reset ngoài ( External Reset )

Một tín hiệu Reset ngoài được phát ra bởi 1 mức thấp ở trong chân Reset . Xung Reset thì dài hơn độ rộng xung cực tiêu (xem bảng 19 ) sẽ phát ra tín hiệu Reset dù là bộ định thời không hoạt động . Các xung ngắn hơn thì không bảo đảm để phát ra tín hiệu Reset . Khi tín hiệu đặt tiền dàn đến ngưỡng điện áp ngưỡng Reset –  $V_{RST}$  ở trên sườn dương của nó , bộ đếm trễ khởi động MCU sau khi chu kỳ thời gian chờ  $t_{TOUT}$  vừa hết hạn .

**Figure 25.** External Reset During Operation

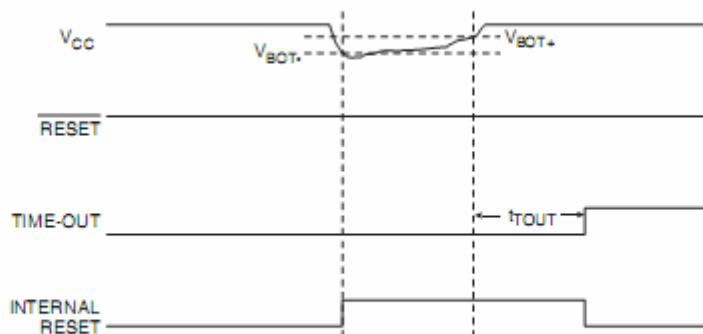
## Sự dò yếu điện áp

Atmega 128 có một mạch dò yếu điện áp ở trên chip để theo dõi mức  $V_{CC}$  trong suốt quá trình hoạt động bằng việc so sánh nó với mức khởi động. Mức khởi động cho BOD có thể được lựa chọn bởi cầu chì BODLEVEL là 2,7 V (BODLEVEL không được lập trình) hoặc 4.0V nếu BODLEVEL được lập trình). Cấp khởi động sẽ có 1 độ trễ để bảo đảm đầu độc lập của bộ dò điện áp thấp. Độ trễ trên cấp dò nên được diễn dịch như là  $V_{BOT+} = V_{BOT} + V_{HYST}/2$  và  $V_{BOT-} = V_{BOT} - V_{HYST}/2$

Mạch BOT có thể được kích hoạt hoặc vô hiệu hóa bằng cầu chì BODEN. Khi mà cầu chì BOD được kích hoạt (BODEN được lập trình), và  $V_{CC}$  giảm về giá trị bên dưới mức khởi động ( $V_{BOT-}$  trong hình 26), Reset yếu điện áp ngay lập tức được hoạt động. Khi mà  $V_{CC}$  tăng trên cấp khởi động ( $V_{BOT+}$  trong hình 26), bộ đếm thời gian trễ khởi động MCU sau chu kỳ thời gian chờ  $t_{TOUT}$  vừa hết hạn

Mạch BOD sẽ chỉ dò 1 sự sụt giảm điện áp  $V_{CC}$  nếu như điện áp ở dưới cấp khởi động lâu hơn  $t_{BOD}$  được đưa ra trong bảng 19

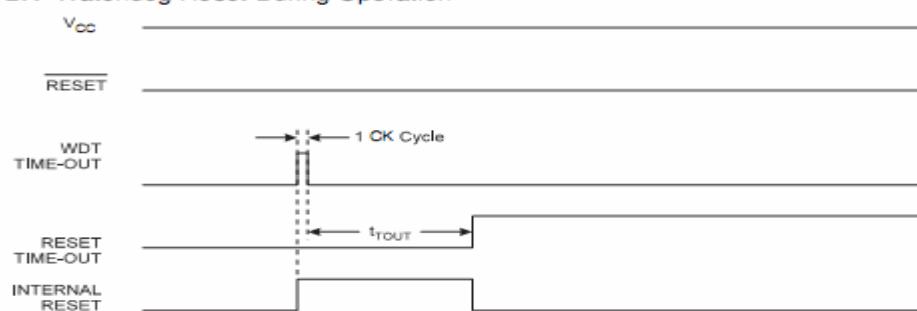
**Figure 26.** Brown-out Reset During Operation



## Reset watchdog

Khi watchdog hết thời gian chờ, nó sẽ phát ra 1 xung Reset ngắn của 1 khoảng thời gian 1 chu kỳ xung nhịp CK.. Trên sườn xuống của xung này thời gian trễ bắt đầu việc đếm chu kỳ thời gian chờ  $t_{TOUT}$ . Tham khảo trang 55 để biết thêm chi tiết về hoạt động của Timer watchdog

**Figure 27.** Watchdog Reset During Operation



## Thanh ghi trạng thái và điều khiển MCU – MCUCSR

Thanh ghi trạng thái và điều khiển MCU cung cấp thông tin về nguồn xung Reset nào là nguyên nhân của một tín hiệu Reset

Bit	7	6	5	4	3	2	1	0	MCUCSR
ReadWrite	JTD	-	-	JTRF	WDRF	BORF	EXTRF	PORF	
Initial Value	0	0	0						
<a href="#">See Bit Description</a>									

Chú ý rằng chỉ EXTRF và PORF là khả dụng ở trong chế độ tương thích với Atmega 103

Bit 4 – JTRF : cờ Reset JTAG

Bit này được cài đặt nếu 1 tín hiệu reset đang bị gây ra bởi một mức logic1 trên thanh ghi Reset JTAG được lựa chọn bởi lệnh JTAG AVR\_RESET . Bit này được Reset bằng 1 tín hiệu Reset bật nguồn ( power on ) , hoặc bằng việc viết mức logic 0 lên cờ

Bit 3 – WDRF : cờ reset Watchdog

Bit này được cài đặt nếu tín hiệu Reset Watchdog xuất hiện . Bit này được Reset bởi 1 tín hiệu Reset bật nguồn ( power – on ) , hoặc viết mức logic 0 lên cờ

Bit 2 – BORF cờ Reset yếu điện áp Brown-out

Bit này được reset nếu 1 tín hiệu reset Brown-out xuất hiện . Bit này được Reset bởi 1 tín hiệu Reset bật nguồn ( power – on ) , hoặc viết mức logic 0 lên cờ

Bit 1 – EXTRF : cờ Reset ngoài

Bit này được cài đặt nếu 1 tín hiệu reset ngoài xuất hiện . Bit này được Reset bởi 1 tín hiệu Reset bật nguồn ( power – on ) , hoặc viết mức logic 0 lên cờ

Bit 0 – PORF : cờ Reset bật nguồn ( Power-on )

Bit này được cài đặt nếu tín hiệu Reset Power-on xuất hiện . bit này được reset chỉ bằng việc viết mức logic 0 lên cờ

Để hiểu cách sử dụng của các cờ Reset với 1 điều kiện Reset giống nhau , người sử dụng nên đọc và sau đó reset thanh ghi MCUCSR để dàng trong lập trình . Nếu như thanh ghi bị xóa trước 1 tín hiệu reset khác xuất hiện , nguồn của tín hiệu reset có thể được tìm thấy bằng việc khảo sát các cờ reset

## Sự tham khảo điện áp bên trong

Atmega 128 có 1 vùng tham khảo ở bên trong (internal bandgap reference) . Điểm chuẩn này được sử dụng cho sự dò yếu điện áp (Brown-out Detection) , và nó có thể được sử dụng như là 1 đầu vào đến bộ so sánh tương tự hoặc bộ chuyển đổi ADC . Mốc chuẩn 2.56 V đến bộ chuyển đổi ADC được sinh ra từ Vùng tham khảo bên trong (internal bandgap reference)

Tín hiệu kích hoạt điện áp tham khảo và thời gian khởi động ( Voltage Reference Enable Signals and Start-up Time )

Chuẩn điện áp có 1 thời gian khởi động cái mà có thể ảnh hưởng đến phương pháp nó nên được sử dụng . Thời gian khởi động được đưa ra trong bảng 20 . Để tiết kiệm điện , điện áp chuẩn thì không luôn luôn được bật . Điện áp chuẩn thì được bật trong suốt các trường hợp dưới đây :

- Khi mà BOD được kích hoạt ( bằng cách lập trình cầu chì BODEN )
- Khi mà vùng điện áp chuẩn được kết nối tới bộ so sánh analog ( bằng việc cài đặt bit ACBG trong thanh ghi ACSR )
- Khi mà bộ chuyển đổi ADC được kích hoạt

Như vậy , khi mà BOD không được kích hoạt , sau khi cài đặt bit ACBG hoặc kích hoạt ADC , người sử dụng phải luôn luôn cho phép mốc chuẩn (reference) để khởi động trước khi đầu ra từ bộ so sánh tương tự hoặc bộ chuyển đổi ADC được sử dụng . Để giảm tốn hao công suất trong chế độ tắt nguồn , người sử dụng có thể tránh 3 điều kiện bên dưới để đảm bảo rằng mốc chuẩn được tắt trước khi truy nhập vào chế độ tắt nguồn Power- down

**Table 20. Internal Voltage Reference Characteristics**

Symbol	Parameter	Min	Typ	Max	Units
$V_{BG}$	Bandgap reference voltage	1.15	1.23	1.40	V
$t_{BG}$	Bandgap reference start-up time		40	70	$\mu s$
$I_{BG}$	Bandgap reference current consumption		10		$\mu A$

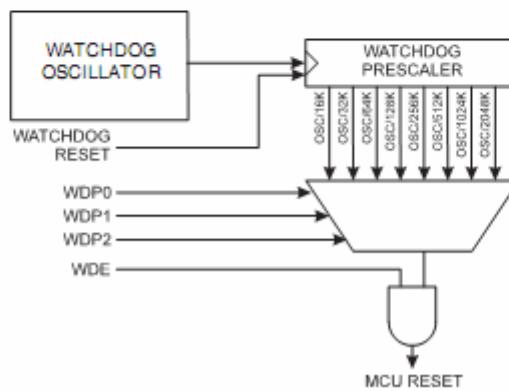
## Timer watchdog

Timer watchdog thì bị khóa từ 1 bộ tạo dao động trên chip riêng biệt cái mà chạy ở 1 MHz . Đây là giá trị điển hình tại  $V_{CC} = 5V$  . Xem dữ liệu mô tả đặc tính cho các giá trị điển hình ở các mức  $V_{CC}$  khác nhau . Bằng việc điều khiển bộ đếm gộp trước của Timer watchdog , khoảng Reset watchdog có thể được điều chỉnh như được chỉ ra trong bảng 22 trang 57 . Bit WDR – Watchdog Reset – hướng dẫn reset Timer watchdog . Timer watchdog cũng được reset khi mà nó bị vô hiệu hóa và khi 1 tín hiệu reset chip xuất hiện. 8 chu kì xung nhịp khác nhau có thể được lựa chọn để xác định chu kì reset . Nếu chu kì reset hết hạn mà không có tín hiệu Reset watchdog khác , Atmega 128 reset và thực thi từ vecto reset . Về chi tiết bộ định thời trên Reset watchdog , tham khảo trang 54

Để ngăn cản sự vô hiệu hóa vô tình của watchdog hoặc vô tình thay đổi chu kì time-out , 3 cấp an toàn khác nhau được lựa chọn bằng cầu chì M103C và WDTON được chỉ ra trong bảng 21 . Mức độ an toàn 0 tương ứng để cài đặt trong Atmega 103 . Không có hạn chế nào trong việc kích hoạt WDT trong bất cứ cấp an toàn nào . Tham khảo “timed Sequences for Changing the Configuration of the watchdog Timer “ ở trang 58 để biết thêm chi tiết

**Table 21.** WDT Configuration as a Function of the Fuse Settings of M103C and WDTON.

M103C	WDTON	Safety Level	WDT Initial State	How to Disable the WDT	How to Change Time-out
Unprogrammed	Unprogrammed	1	Disabled	Timed sequence	Timed sequence
Unprogrammed	Programmed	2	Enabled	Always enabled	Timed sequence
Programmed	Unprogrammed	0	Disabled	Timed sequence	No restriction
Programmed	Programmed	2	Enabled	Always enabled	Timed sequence

**Figure 28.** Watchdog Timer

### Thanh ghi điều khiển Timer watchdog – WDTCR

Bit	7	6	5	4	3	2	1	0	WDTCR
Read/Write	-	-	-	WDCE	WDE	WDP2	WDP1	WDP0	
Initial Value	R	R	R	R/W	R/W	R/W	R/W	R/W	0 0 0 0 0 0 0 0 0

Bit 7..5 – Res : các bit dự trữ

Các bit này là các bit dự trữ trong Atmega 128 và luôn luôn được đọc là 0 .

Bit 4 – WDCE : kích hoạt thay đổi watchdog

Bit này phải được cài đặt khi mà bit WDE được ghi mức logic 0 . Nói cách khác , watchdog sẽ không bị vô hiệu hóa . mỗi lần viết là 1 , phần cứng sẽ xóa bit này sau 4 chu kì xung nhịp. Tham khảo phần miêu tả của bit WDE cho về thủ tục vô hiệu hóa 1 watchdog .Trong các cấp an toàn 1 và 2 bit này cũng phải được cài đặt khi mà sự thay đổi các bit của bộ đếm gộp trước (prescaler ) . Xem thêm phần “timed Sequences for Changing the Configuration of the watchdog Timer) trang 58 để biết thêm chi tiết

Bit 3 – WDE : kích hoạt watchdog

Khi mà WDE được viết là 1, Timer watchdog được kích hoạt , và nếu WDE được viết mức logic 0 , thì các chức năng của Timer watchdog bị vô hiệu hóa . WDE có

thể chỉ bị xóa nếu như WDCE có mức logic là 1 . Để vô hiệu hóa 1 Timer watchdog được kích hoạt , các quy trình dưới đây phải được tuân theo

1. trong sự hoạt động giống nhau , viết mức logic 1 lên WDCE và WDE . 1 mức logic 1 phải được viết lên WDE cho dù nó được cài đặt trước khi hoạt động vô hiệu hóa bắt đầu
2. Trong vòng 4 chu kì xung nhịp kế tiếp , viết mức logic 0 lên WDE . Điều này vô hiệu hóa watchdog

Trong cấp an toàn 2 , nó không thể thực hiện để vô hiệu hóa Timer watchdog , thậm chí với thuật toán được mô tả bên dưới . xem “ timed Sequences for Changing the Configuration of the watchdog Timer” ở trang 58 .

Bit 2..0 – WDP2, WDP1 , WDP0 : Bộ đếm gộp trước 2,1,0 của Timer watchdog

Các bit WDP2, WDP1 , WDP0 xác định việc đếm gộp trước của Timer watchdog khi mà Timer watchdog được kích hoạt . Giá trị việc đếm gộp trước khác nhau và các chu kì thời gian chờ tương ứng của chúng được chỉ ra trong bảng 22

**Table 22. Watchdog Timer Prescale Select**

WDP2	WDP1	WDP0	Number of WDT Oscillator Cycles	Typical Time-out at $V_{CC} = 3.0V$	Typical Time-out at $V_{CC} = 5.0V$
0	0	0	16K (16,384)	14.8 ms	14.0 ms
0	0	1	32K (32,768)	29.6 ms	28.1 ms
0	1	0	64K (65,536)	59.1 ms	56.2 ms
0	1	1	128K (131,072)	0.12 s	0.11 s
1	0	0	256K (262,144)	0.24 s	0.22 s
1	0	1	512K (524,288)	0.47 s	0.45 s
1	1	0	1,024K (1,048,576)	0.95 s	0.9 s
1	1	1	2,048K (2,097,152)	1.9 s	1.8 s

Đoạn code ví dụ dưới đây chỉ ra 1 hàm assembly và C cho việc tắt WDT . Ví dụ này giả định rằng các ngắt đã được điều khiển ( ví dụ bằng cách vô hiệu hóa các ngắt chung ) vì vậy sẽ không có ngắt nào xuất hiện trong suốt quá trình thực thi các hàm này .

**Assembly Code Example**

```

WDT_off:
; Reset WDT
wdr
in r16, WDTCR
; Write logical one to WDCE and WDE
ori r16, (1<<WDCE) | (1<<WDE)
out WDTCR, r16
; Turn off WDT
ldi r16, (0<<WDE)
out WDTCR, r16
ret

```

**C Code Example**

```

void WDT_off(void)
{
    /* Reset WDT*/
    __watchdog_reset();
    /* Write logical one to WDCE and WDE */
    WDTCR |= (1<<WDCE) | (1<<WDE);
    /* Turn off WDT */
    WDTCR = 0x00;
}

```

**Các chuỗi được định thời cho việc thay đổi cấu hình của Timer watchdog**

Các chuỗi cho việc thay đổi cấu hình khác biệt không đáng kể giữa 3 cấp an toàn . Phương pháp phân chia thì được miêu tả cho mỗi cấp

**Cấp an toàn 0**

Chế độ này thì tương thích với hoạt động của Timer watchdog được tìm thấy trong Atmega 103 . Timer watchdog thì bị vô hiệu hóa từ đầu , nhưng không thể được kích hoạt bằng việc ghi bit WDE là 1 mà không có bất cứ sự hạn chế nào . Chu kỳ thời gian chờ có thể bị thay đổi tại bất cứ thời gian nào giới hạn . Để vô hiệu hóa một 1 Timer watchdog đã kích hoạt , thì quy trình được miêu tả ở trang 56 (sự miêu tả Bit WDE) phải được tuân theo .

**Cấp an toàn 1**

Trong chế độ này , Timer watchdog bị vô hiệu hóa từ đầu , nhưng có thể được kích hoạt bằng việc viết bit WDE là 1 dù cho bất cứ sự ngăn cản nào . 1 chuỗi được định thời là cần thiết khi việc thay đổi chu kỳ thời gian chờ của watchdog hoặc việc vô hiệu hóa1 Timer watchdog đã kích hoạt. Để vô hiệu hóa 1 Timer watchdog đã kích hoạt,và/hoặc thay đổi thời gian chờ watchdog , các quy trình dưới đây phải được tuân theo :

1. trong hoạt động giống nhau , viết mức logic 1 lên bit WDCE và WDE . Một mức logic 1 phải được ghi bất chấp giá trị trước của bit WDE
2. Trong vòng 4 chu kì xung nhịp kế tiếp , trong các quá trình điều khiển giống nhau , viết các bit WDE và WDP như ý muốn nhưng với bit WDCE đã được xóa

## Cấp an toàn 2

Trong chế độ này , Timer watchdog luôn luôn được kích hoạt , và bit WDE sẽ luôn luôn được đọc như là 1 . một chuỗi được định thời là cần thiết khi thay đổi chu kì thời gian chờ của watchdog . Để thay đổi thời gian chờ của watchdog , các quy trình dưới đây phải được tuân theo :

1. Trong các quá trình điều khiển giống nhau , viết mức logic 1 lên WDCE và WDE . Cho dù WDE luôn được cài đặt , WDE phải được viết là 1 để khởi động chuỗi được định thời
2. Trong vòng 4 chu kì xung nhịp kế tiếp , trong các hoạt động giống nhau , viết các bit WDP như giá trị mong muốn , nhưng với bit WDCE đã được xóa. Giá trị được viết lên bit WDE thì không thích hợp

## VIII . Các ngắt

Phần này miêu tả các đặc trưng của việc sử lý ngắt như được chạy trong Atmega 128 . Để cho một sự diễn tả chung về việc sử lý các ngắt của AVR , tham khảo phần “Reset và sử lý ngắt “ ở trang 15

### Các vecto ngắt trong Atmega 128

Table 23. Reset and Interrupt Vectors

Vector No.	Program Address <sup>(1)</sup>	Source	Interrupt Definition
1	\$0000 <sup>(1)</sup>	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	\$0002	INT0	External Interrupt Request 0
3	\$0004	INT1	External Interrupt Request 1
4	\$0006	INT2	External Interrupt Request 2
5	\$0008	INT3	External Interrupt Request 3
6	\$000A	INT4	External Interrupt Request 4
7	\$000C	INT5	External Interrupt Request 5
8	\$000E	INT6	External Interrupt Request 6
9	\$0010	INT7	External Interrupt Request 7
10	\$0012	TIMER2 COMP	Timer/Counter2 Compare Match
11	\$0014	TIMER2 OVF	Timer/Counter2 Overflow
12	\$0016	TIMER1 CAPT	Timer/Counter1 Capture Event
13	\$0018	TIMER1 COMPA	Timer/Counter1 Compare Match A
14	\$001A	TIMER1 COMPB	Timer/Counter1 Compare Match B
15	\$001C	TIMER1 OVF	Timer/Counter1 Overflow
16	\$001E	TIMER0 COMP	Timer/Counter0 Compare Match
17	\$0020	TIMER0 OVF	Timer/Counter0 Overflow
18	\$0022	SPI, STC	SPI Serial Transfer Complete
19	\$0024	USART0, RX	USART0, Rx Complete
20	\$0026	USART0, UDRE	USART0 Data Register Empty
21	\$0028	USART0, TX	USART0, Tx Complete
22	\$002A	ADC	ADC Conversion Complete
23	\$002C	EE READY	EEPROM Ready
24	\$002E	ANALOG COMP	Analog Comparator
25	\$0030 <sup>(3)</sup>	TIMER1 COMPC	Timer/Counter1 Compare Match C
26	\$0032 <sup>(3)</sup>	TIMER3 CAPT	Timer/Counter3 Capture Event
27	\$0034 <sup>(3)</sup>	TIMER3 COMPA	Timer/Counter3 Compare Match A
28	\$0036 <sup>(3)</sup>	TIMER3 COMPB	Timer/Counter3 Compare Match B
29	\$0038 <sup>(3)</sup>	TIMER3 COMPC	Timer/Counter3 Compare Match C
30	\$003A <sup>(3)</sup>	TIMER3 OVF	Timer/Counter3 Overflow

**Table 23.** Reset and Interrupt Vectors (Continued)

Vector No.	Program Address <sup>(2)</sup>	Source	Interrupt Definition
31	\$003C <sup>(3)</sup>	USART1, RX	USART1, Rx Complete
32	\$003E <sup>(3)</sup>	USART1, UDRE	USART1 Data Register Empty
33	\$0040 <sup>(3)</sup>	USART1, TX	USART1, Tx Complete
34	\$0042 <sup>(3)</sup>	TWI	Two-wire Serial Interface
35	\$0044 <sup>(3)</sup>	SPM READY	Store Program Memory Ready

- Notes:
1. When the BOOTRST fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming" on page 273.
  2. When the IVSEL bit in MCUCR is set, interrupt vectors will be moved to the start of the Boot Flash section. The address of each interrupt vector will then be address in this table added to the start address of the boot Flash section.
  3. The Interrupts on address \$0030 - \$0044 do not exist in ATmega103 compatibility mode.

Bảng 24 chỉ ra sự bố trí các vecto ngắt và Reset cho sự kết hợp khác nhau của việc cài đặt BOOTRST và IVSEL. Nếu chương trình không bao giờ kích hoạt 1 nguồn ngắt , các vecto ngắt thì không được sử dụng , 1 đoạn code chương trình thông thường có thể được đặt vào trong các vị trí này . Đây cũng là trường hợp nếu vecto Reset ở trong đoạn chương trình ứng dụng trong khi các vecto ngắt thì ở trong khu vực khởi động Boot section hoặc vice versa

**Table 24.** Reset and Interrupt Vectors Placement

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	\$0000	\$0002
1	1	\$0000	Boot Reset Address + \$0002
0	0	Boot Reset Address	\$0002
0	1	Boot Reset Address	Boot Reset Address + \$0002

Note: The Boot Reset Address is shown in Table 112 on page 284. For the BOOTRST fuse "1" means unprogrammed while "0" means programmed.

Các cài đặt chương trình chung và thông dụng nhất cho việc đặt địa chỉ các vecto ngắt và vecto Reset của Atmega 128 là

```

-----+----+
Address LabelsCode      Comments
-----+----+
$0000  jmp   RESET    ; Reset Handler
$0002  jmp   EXT_INT0 ; IRQ0 Handler
$0004  jmp   EXT_INT1 ; IRQ1 Handler
$0006  jmp   EXT_INT2 ; IRQ2 Handler
$0008  jmp   EXT_INT3 ; IRQ3 Handler
$000A  jmp   EXT_INT4 ; IRQ4 Handler
$000C  jmp   EXT_INT5 ; IRQ5 Handler
$000E  jmp   EXT_INT6 ; IRQ6 Handler
$0010  jmp   EXT_INT7 ; IRQ7 Handler
$0012  jmp   TIM2_COMP ; Timer2 Compare Handler
$0014  jmp   TIM2_OVF  ; Timer2 Overflow Handler
$0016  jmp   TIM1_CAPT ; Timer1 Capture Handler
$0018  jmp   TIM1_COMPA; Timer1 CompareA Handler
$001A  jmp   TIM1_COMPAREB; Timer1 CompareB Handler
$001C  jmp   TIM1_OVF  ; Timer1 Overflow Handler
$001E  jmp   TIM0_COMP ; Timer0 Compare Handler
$0020  jmp   TIM0_OVF  ; Timer0 Overflow Handler
$0022  jmp   SPI_STC  ; SPI Transfer Complete Handler
$0024  jmp   USART0_RXC; USART0 RX Complete Handler
$0026  jmp   USART0_DRE; USART0_UDR Empty Handler
$0028  jmp   USART0_TXC; USART0 TX Complete Handler
$002A  jmp   ADC       ; ADC Conversion Complete Handler
$002C  jmp   EEPROM_RDY; EEPROM Ready Handler
$002E  jmp   ANA_COMP  ; Analog Comparator Handler
$0030  jmp   TIM1_COMPAREC; Timer1 CompareC Handler
$0032  jmp   TIM3_CAPT  ; Timer3 Capture Handler
$0034  jmp   TIM3_COMPA; Timer3 CompareA Handler
$0036  jmp   TIM3_COMPAREB; Timer3 CompareB Handler
$0038  jmp   TIM3_COMPAREC; Timer3 CompareC Handler
$003A  jmp   TIM3_OVF  ; Timer3 Overflow Handler
$003C  jmp   USART1_RXC; USART1 RX Complete Handler
$003E  jmp   USART1_DRE; USART1_UDR Empty Handler
$0040  jmp   USART1_TXC; USART1 TX Complete Handler
$0042  jmp   TWI       ; Two-wire Serial Interface Interrupt Handler
$0044  jmp   SPM_RDY   ; SPM Ready Handler
:
$0046  RESET:ldir16, high(RAMEND); Main program start
$0047  out   SPM,r16  ; Set stack pointer to top of RAM
$0048  ldi   r16, low(RAMEND)
$0049  out   SPL,r16
$004A  sei   ; Enable interrupts
$004B  <instr> xxxx
***   ***   ***

```

Khi mà cầu chì BOOTRST không được lập trình, kích cỡ của khu vực khởi động được cài đặt lên 8K bytes và bit IVSEL trong thanh ghi MCUCR được cài đặt trước khi bắt cứ ngắt nào được kích hoạt, các cài đặt chung và thông dụng nhất cho các địa chỉ vecto ngắt và Reset là

```

Address LabelsCode      Comments
$0000  RESET:ldi    r16,high(RAMEND); Main program start
$0001      out     SPH,r16   ; Set stack pointer to top of RAM
$0002      ldi     r16,low(RAMEND)
$0003      out     SPL,r16
$0004      sei          ; Enable interrupts
$0005      <instr>  xxxx
;
.org $F002
$F002      jmp     EXT_INT0  ; IRQ0 Handler
$F004      jmp     EXT_INT1  ; IRQ1 Handler
...
$F044      jmp     SPM_RDY   ; Store Program Memory Ready Handler

```

Khi cầu chì BOOTRST được lập trình và độ lớn khu vực khởi động cài đặt lên 8K bytes , cài đặt chương trình chung và thông dụng nhất cho các địa chỉ của Vecto ngắt và Reset là

```

Address LabelsCode      Comments
.org $0002
$0002      jmp     EXT_INT0  ; IRQ0 Handler
$0004      jmp     EXT_INT1  ; IRQ1 Handler
...
$0044      jmp     SPM_RDY   ; Store Program Memory Ready Handler
;
.org $F000
$F000  RESET: ldi    r16,high(RAMEND); Main program start
$F001      out     SPH,r16   ; Set stack pointer to top of RAM
$F002      ldi     r16,low(RAMEND)
$F003      out     SPL,r16
$F004      sei          ; Enable interrupts
$F005      <instr>  xxxx

```

Khi mà cầu chì BOOTRST được lập trình , kích cỡ khu vực khởi động đặt lên 8K bytes và bit IVSEL trong thanh ghi MCUCR được cài đặt trước khi các ngắt được kích hoạt , cài đặt chương trình chung và điển hình nhất cho các địa chỉ các vecto ngắt và Reset là

Address	Labels	Code	Comments
;			
.org \$F000			
\$F000	jmp	RESET	; Reset handler
\$F002	jmp	EXT_INT0	; IRQ0 Handler
\$F004	jmp	EXT_INT1	; IRQ1 Handler
...	...	...	;
\$F044	jmp	SPM_RDY	; Store Program Memory Ready Handler
\$F046	RESET: ldi	r16,high(RAMEND)	; Main program start
\$F047	out	SPH,r16	; Set stack pointer to top of RAM
\$F048	ldi	r16,low(RAMEND)	
\$F049	out	SPL,r16	
\$F04A	sei		; Enable interrupts
\$F04B	<instr>	xxxx	

Việc di chuyển các ngắt giữa các chương trình ứng dụng và không gian khởi động ( Moving Interrupts Between Application and Boot Space )

Thanh ghi điều khiển các ngắt chung điều khiển việc sắp xếp vị trí của bảng các vecto ngắt .

### Thanh ghi điều khiển MCU – MCUCR

Bit	7	6	5	4	3	2	1	0	MCUCR
Read/Write	SRE	SRW10	SE	SM1	SM0	SM2	IVSEL	IVCE	
Initial Value	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	0	0	0	0	0	0	0	0	

Bit 1 – IVSEL : lựa chọn các vecto ngắt

Khi mà bit IVSEL bị xóa (0) , các vecto ngắt được đặt vào vị trí bắt đầu của bộ nhớ Flash . Khi bit này được đặt là 1 , các vecto ngắt được di chuyển đến vị trí bắt đầu của khu vực tải chế độ khởi động ở trong bộ nhớ Flash . Địa chỉ hiện thời của vị trí bắt đầu của khu vực khởi động trong bộ nhớ Flash (Boot Flash section ) được xác định bởi các câu lệnh BOOTSZ . tham khảo phần “Boot Loader Support ” và “Read-while- write Self-Programming ” ở trang 273 để thêm chi tiết . Để tránh các thay đổi vô tình của các bảng các vec to ngắt , 1 quy trình ghi đặc biệt phải được tuân theo để thay đổi bit IVSEL :

1. Viết bit kích hoạt thay đổi vecto ngắt (IVCE) là 1
2. Trong vòng 4 chu kỳ xung nhịp , viết giá trị mong muốn lên IVSEL trong khi viết là 0 lên bit IVCE

Các ngắt sẽ bị vô hiệu hóa 1 cách tự động trong khi chuỗi này được thi hành , các ngắt bị vô hiệu hóa trong chu kỳ IVCE được cài đặt . và phần dư của chúng bị vô hiệu hóa cho đến sau khi lệnh bên dưới được viết vào IVSEL. Nếu như IVSEL không được ghi , các ngắt còn lại bị vô hiệu hóa trong 4 chu kỳ xung nhịp . Bít I trong thanh ghi trạng thái thì không bị tác động bởi việc vô hiệu hóa tự động

Chú ý rằng : Nếu các vecto ngắt được đặt ở trong khu vực quá trình khởi động và bit khóa quá trình khởi động BLB02 được lập trình , các ngắt sẽ bị vô hiệu hóa trong quá trình thực thi từ khu vực ứng dụng . Nếu các vecto ngắt được đặt trong khu vực chương trình ứng dụng và các bit khóa Boot BLB02 được lập trình, các ngắt bị vô hiệu hóa trong khi quá trình thực thi từ khu vực tải quá trình khởi động . tham khảo thêm phần “Boot Loader Support ” ở trang 273 để biết thêm chi tiết về các bit khóa BOOT

Bit 0 – IVCE : kích hoạt thay đổi các vecto ngắt

Bít IVCE phải được ghi lên mức logic 1 để kích hoạt thay đổi của bit IVSEL. IVCE thì bị xóa bằng phần cứng 4 chu kỳ xung nhịp sau khi nó được viết hoặc khi IVSEL được ghi . Việc cài đặt bit IVCE sẽ vô hiệu hóa các ngắt , như là được diễn tả trong phần miêu tả IVSEL bên trên . Xem đoạn code ví dụ bên dưới

**Assembly Code Example**

```
Move_interrupts:  
    ; Enable change of interrupt vectors  
    ldi r16, (1<<IVCE)  
    out MCUCR, r16  
    ; Move interrupts to boot flash section  
    ldi r16, (1<<IVSEL)  
    out MCUCR, r16  
    ret
```

**C Code Example**

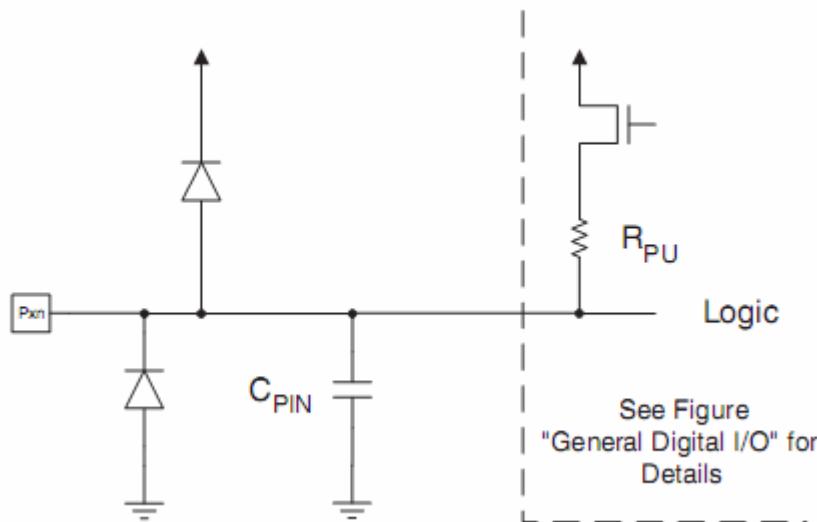
```
void Move_interrupts(void)  
{  
    /* Enable change of interrupt vectors */  
    MCUCR = (1<<IVCE);  
    /* Move interrupts to boot flash section */  
    MCUCR = (1<<IVSEL);  
}
```

## IX . Các cổng vào ra ( I/O port )

### Giới thiệu

Tất cả các cổng vào của AVR đều có chức năng đọc-sửa đổi-ghi khi được sử dụng như là các cổng I/O digital chung . Điều này có nghĩa là hướng của 1 chân cổng có thể được thay đổi mà có sự thay đổi hướng vô tình nào của bất cứ chân nào khác với lệnh SBI và CBI . Cái giống như vậy được áp dụng khi thay đổi giá trị điều khiển (nếu được cấu hình như là đầu ra ) hoặc việc kích hoạt/vô hiệu hóa của điện trở kéo lên (nếu được cấu hình như là đầu vào ) . Mỗi bộ đệm đầu ra có đặc tính điều khiển đối xứng với cả hai tần nhiệt cao và nguồn điện dung. Bộ điều khiển chân thì đủ mạnh để điều khiển bộ hiển thị LED 1 cách trực tiếp . Tất cả các chân cổng có điện trở kéo lên có khả năng lựa chọn riêng biệt với 1 trở kháng bắt biến của điện áp nguồn cấp . Tất cả các chân I/O có các diode bảo vệ để cả 2 V<sub>CC</sub> và chân Ground như là được hiển thị trong hình 29 . Tham khảo “electrical characteristic” ở trang 318 để có bảng các tham số đầy đủ .

**Figure 29.** I/O Pin Equivalent Schematic



Tất cả các thanh ghi và các bit tham khảo ở trong phần này được ghi trong 1 mẫu chung .

Trong một trường hợp thấp hơn “x” đại diện cho numbering letter của cổng , và 1 két dưới (lower case ) “n” đại diện cho số thứ tự bit . Tuy nhiên khi việc sử dụng các thanh ghi hoặc các bit xác định trong 1 chương trình , mẫu chính xác phải được sử dụng . Ví dụ PORTB3 cho bit số 3 trong cổng B , được dẫn chứng chung như là PORTxn . Thanh ghi I/O vật lí và vị trí các bit được liệt kê trong “Register Description for I/O Ports ” ở trang 87.

Ba vị trí địa chỉ bộ nhớ I/O thì được phân phối cho mỗi cổng , 1 cho thanh ghi dữ liệu – PORTx , Thanh ghi định hướng dữ liệu – DDRx , và các chân đầu vào của cổng – PINx. Vị trí I/O của đầu vào các chân thì chỉ được đọc , trong khi thanh ghi dữ liệu và thanh ghi định hướng dữ liệu được đọc và ghi .Thêm vào đó , bit vô hiệu hóa dùng lại

– PUD trong SFIOR vô hiệu hóa chức năng pull-up của cho tất cả các chân trong tất cả các cổng khi được cài đặt.

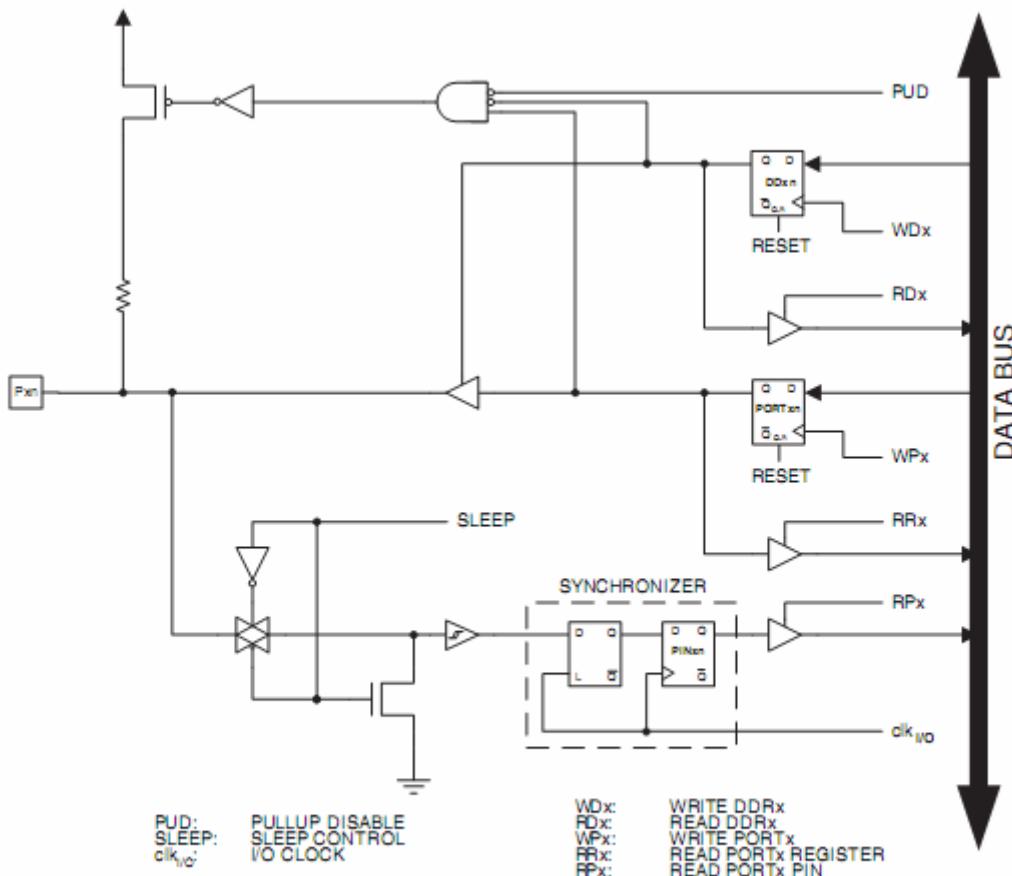
Việc sử dụng các cổng vào ra như là các cổng vào ra số chung được miêu tả trong “Ports as general Digital I/O” ở trang 67 . Hầu hết các chân cổng là đa hợp với các hàm luân phiên (alternate functions) cho các đặc tính ngoại vi trên thiết bị . Cách mà mỗi hàm luân phiên gây nhiễu với các chân cổng thì được miêu tả trong “alternate port functions” trên trang 71 . Tham khảo thêm phần module riêng biệt cho mỗi cho 1 sự miêu tả đầy đủ của các hàm luân phiên (Alternate function)

Chú ý rằng việc kích hoạt hàm luân phiên của vài chân cổng thì không ảnh hưởng đến việc sử dụng các chân khác trong cổng như là các cổng I/O số chung .

### Các cổng như là I/O kĩ thuật số chung ( Ports as General Digital I/O)

Các cổng là các cổng I/O 2 hướng với xung lựa chọn bên trong (optional internal pull-up) . Hình 30 chỉ ra 1 sự miêu tả chức năng của 1 chân cổng I/O , được gọi chung là Pxн

**Figure 30. General Digital I/O<sup>17</sup>**



Note: 1. WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. clk<sub>I/O</sub>, SLEEP, and PUD are common to all ports.

## Cấu hình các chân

Mỗi chân cổng bao gồm 3 bit đăng ký : DD<sub>xn</sub> , PORT<sub>xn</sub> , PIN<sub>xn</sub> . Như là được chỉ ra trong “ Register Description for I/O ports ‘ ở trang 87. Các bit DD<sub>xn</sub> được truy nhập tại địa chỉ I/O DDRx , các bit PORT<sub>xn</sub> tại địa chỉ I/O PORTx và các bit PIN<sub>xn</sub> tại địa chỉ I/O PINx

Bit DD<sub>xn</sub> trong thanh ghi DDRx lựa chọn hướng của chân này . Nếu DD<sub>xn</sub> được viết mức logic 0 , Px<sub>n</sub> được cấu hình là như 1 chân đầu vào.

Nếu PORT<sub>xn</sub> được viết mức logic 1 khi chân này được cấu hình như là 1 chân đầu vào , 1 điện trở pull-up được kích hoạt . Để tắt điện trở pull-up , PORT<sub>xn</sub> phải được viết là mức logic 0 hoặc chân phải được cấu hình như là 1 chân đầu ra . Các chân cổng thì có 3 trạng thái khi mà điều kiện Reset trở nên hoạt động , dù cho không có bộ định thời nào đang chạy .

Nếu PORT<sub>xn</sub> được viết mức 1 khi mà chân này được cấu hình như là 1 chân đầu ra , chân cổng được điều khiển mức cao (1) . Nếu PORT<sub>xn</sub> được viết mức logic 0 khi mà chân được cấu hình như là chân đầu ra , chân cổng được điều khiển ở mức thấp (0)

Khi bật tắt giữa 3 trạng thái ({DD<sub>xn</sub> , PORT<sub>xn</sub>})=0b00 ) và đầu ra cao ({DD<sub>xn</sub> , PORT<sub>xn</sub> } = 0b11 , một trạng thái trung gian với pull-up đã kích hoạt ({DD<sub>xn</sub> , PORT<sub>xn</sub> }=0b01 ) hoặc đầu ra thấp ({DD<sub>xn</sub>,PORT<sub>xn</sub>}=0b10 ) phải xuất hiện . Thông thường , pull-up đã kích hoạt trạng thái thì có thể truy nhập đầy đủ , như là một môi trường có trở kháng cao sẽ không nhận biết sự khác nhau giữa 1 bộ điều khiển mạnh và một pull-up . Nếu đây không phải là trường hợp này , bit PUD trong thanh ghi SFIOR có thể được viết là 1 để vô hiệu hóa tất cả các pull-up trong tất cả các cổng .

Việc chuyển đổi giữa cổng vào với pull-up và cổng ra thấp làm nảy sinh các vấn đề giống nhau . Người sử dụng phải sử dụng 1 trong 3 trạng thái({DD<sub>xn</sub> , PORT<sub>xn</sub> } = 0b00 ) hoặc trạng thái đầu ra mức cao ({DD<sub>xn</sub> , PORT<sub>xn</sub> }=0b11 ) như là các bước trung gian

Bảng 25 tóm tắt các tín hiệu điều khiển cho giá trị của chân

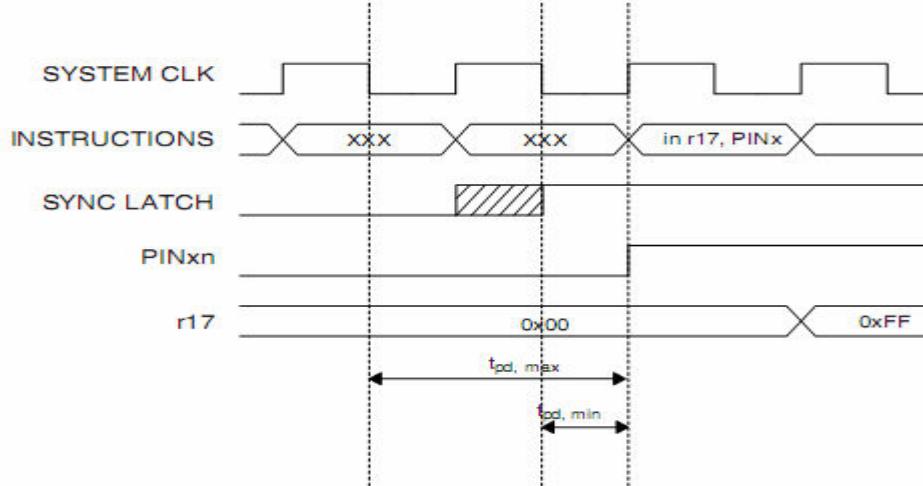
Table 25. Port Pin Configurations

DD <sub>xn</sub>	PORT <sub>xn</sub>	PUD (in SFIOR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

## Đọc giá trị các chân

Không phụ thuộc vào việc cài đặt các bit định hướng dữ liệu DD<sub>Xn</sub> , các chân cổng có thể được đọc thông qua bit thanh ghi PIN<sub>Xn</sub> . Như được chỉ ra trong hình 30 , bit thanh ghi PIN<sub>Xn</sub> và then cài trước cấu tạo 1 cách đồng bộ . Điều này là cần thiết để tránh tính nửa bền nếu như các chân vật lí thay đổi giá trị gần sườn của xung nhịp bên trong (internal clock ) , nhưng nó cũng giới thiệu 1 độ trễ . Hình 31 chỉ ra 1 mạch bộ định thời của sự đồng bộ khi việc đọc 1 giá trị chân được đặt ở bên ngoài . Giá trị cực đại và cực tiểu của sự trễ lan truyền thì được kí hiệu tương ứng là  $t_{pd,max}$  và  $t_{pd,min}$

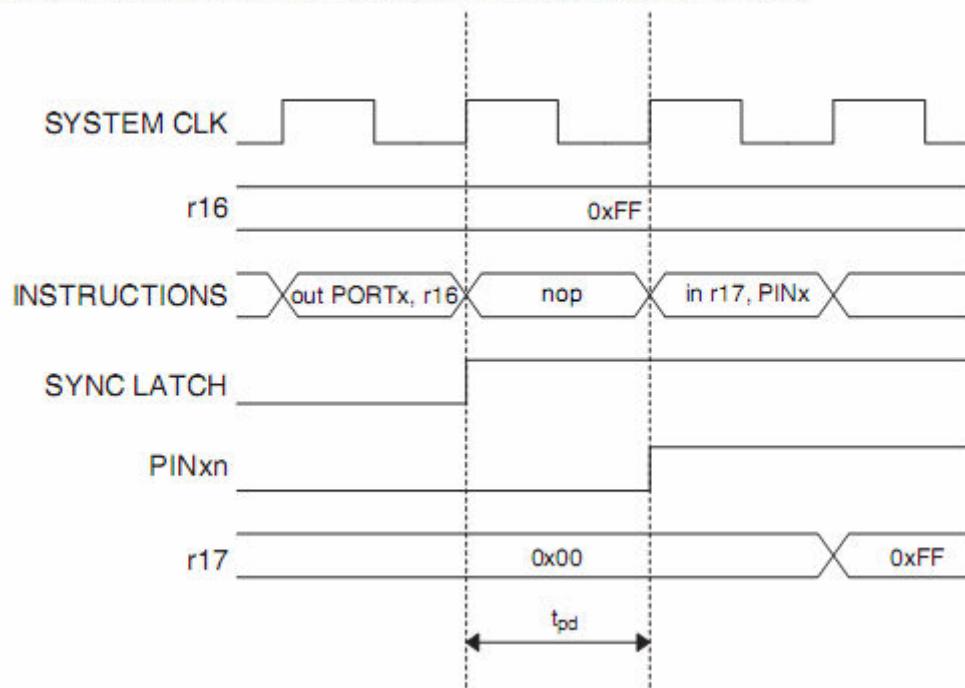
**Figure 31. Synchronization when Reading an Externally Applied Pin Value**



Coi như chu kì xung nhịp được bắt đầu ngắn sau khi sườn xuống đầu tiên của xung nhịp hệ thống . Then cài (latch) thì được đóng khi mà xung nhịp thấp , và trở thành trong suốt khi xung nhịp ở mức cao , như là được hiển thị bằng khoảng bóng của tín hiệu “SYNC LATCH ” . Giá trị tín hiệu thì được chốt (latched) khi mà xung nhịp hệ thống xuống mức thấp . Nó bị khóa bên trong thanh ghi PIN<sub>Xn</sub> tại vị trí của sườn xung nhịp dương kế tiếp . Như là được hiển thị bằng 2 dòng  $t_{pd,max}$  và  $t_{pd,min}$  , một tín hiệu chuyển tiếp đơn trên chân sẽ bị trễ giữa  $\frac{1}{2}$  và  $1\frac{1}{2}$  của chu kì xung nhịp hệ thống phụ thuộc vào thời gian trên của sự xác nhận .

Khi việc đọc lại 1 phần mềm đã xác nhận giá trị của các chân , 1 lệnh nop phải được chèn vào như là được hiện thị trong hình 32 . Lệnh Out cài đặt tín hiệu “SYNC LATCH ” tại sườn dương của xung nhịp . Trong trường hợp này , thời gian trễ  $t_{pd}$  thông qua bộ đồng bộ hóa là 1 chu kì xung nhịp hệ thống .

Figure 32. Synchronization when Reading a Software Assigned Pin Value



Đoạn mã mẫu dưới đây chỉ ra cách để cài đặt các chân 0 và 1 cao và 2,3 thấp của cổng B , và xác định các chân cổng từ 4 đến 7 như là đầu vào với các pull-up được gán vào các chân 6 và 7. Kết quả các giá trị của chân được đọc trở lại , nhưng như là thảo luận từ trước , một lệnh nop thì bao gồm các giá trị hiện thời có thể đọc trở lại được gán đến một vài chân

#### Assembly Code Example<sup>(1)</sup>

```
...
; Define pull-ups and set outputs high
; Define directions for port pins
ldi r16, (1<<PB7) | (1<<PB6) | (1<<PB1) | (1<<PB0)
ldi r17, (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0)
out PORTB, r16
out DDRB, r17
; Insert nop for synchronization
nop
; Read port pins
in r16, PINB
...

```

#### C Code Example<sup>(1)</sup>

```
unsigned char i;
...
/* Define pull-ups and set outputs high */
/* Define directions for port pins */
PORTB = (1<<PB7) | (1<<PB6) | (1<<PB1) | (1<<PB0);
DDRB = (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0);
/* Insert nop for synchronization*/
__no_operation();
/* Read port pins */
i = PINB;
...

```

Note: 1. For the assembly program, two temporary registers are used to minimize the time from pull-ups are set on pins 0, 1, 6, and 7, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

## Kích hoạt đầu vào kĩ thuật số và các chế độ ngủ ( Digital Input Enable and Sleep Modes )

Như là được chỉ ra trong hình 30 , tín hiệu đầu vào số có thể được kẹp (clamped) với đất ở tại đầu vào của mạch khởi động schmitt (schmitt –trigger) . Tín hiệu được kí hiệu SLEEP trong hình vẽ , được cài đặt bằng bộ điều khiển chế độ ngủ MCU (MCU sleep controller ) trong chế độ tắt nguồn , chế độ tiết kiệm nguồn , chế độ chờ Standby

mode , và chế độ chờ mở rộng Extended Standby , để tránh tổn hao điện áp cao nếu một vài tín hiệu đầu vào không được nối đất (left floating ) , hoặc có một tín hiệu tương tự analog đóng lên chân  $V_{CC}/2$

SLEEP được ghi đè lên các chân cổng đã kích hoạt như là các chân ngắt ngoài (External Interrupt) . Nếu các truy vấn ngắt ngoài không được kích hoạt , SLEEP cũng được kích hoạt cho các chân này . SLEEP thì cũng được ghi đè bằng nhiều hàm chức năng khác nhau như là được miêu tả trong “ Alternate Port Function “ trên trang 71 .

Nếu như mức logic cao (1) được đưa ra trong 1 chân ngắt ngoài dị bộ được cấu hình như là “ sờn lên trên ngắt , sờn xuống trên ngắt , hoặc bất cứ sự thay đổi logic nào trên chân “ trong khi ngắt ngoài thì không được kích hoạt , sự tương ứng của các cò báu ngắt ngoài sẽ được cài đặt khi tiếp tục từ các chế độ ngủ được nói đến ở trên , như là kèm trong các chế độ ngủ gây ra truy vấn các thay đổi logic

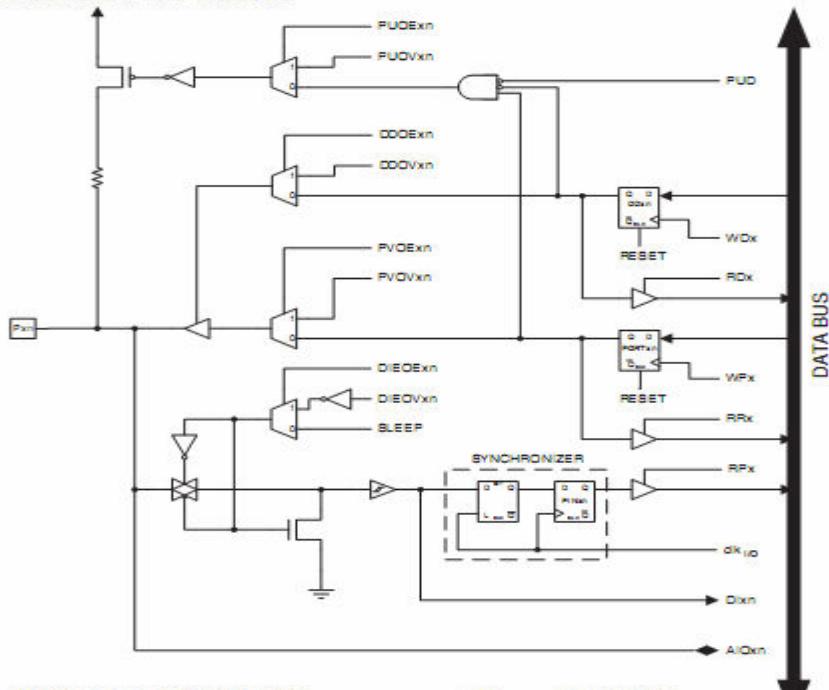
### Các chân không được kết nối

Nếu một vài chân không được sử dụng , nó được khuyến cáo để đảm bảo rằng các chân này có một cấp xác định . Mặc dù tất cả các đầu vào số bị vô hiệu hóa trong các chế độ ngủ như là được miêu tả ở trên , việc nối đất các đầu vào nên được tránh để giảm các dòng điện tổn hao trong tất cả các chế độ khác nhau nơi mà các đầu vào số được kích hoạt (Reset , Active mode và Idle mode)

Phương pháp đơn giản nhất để đảm bảo 1 cấp xác định của 1 chân không sử dụng , là để kích hoạt pull-up bên trong . Trong trường hợp này , pull-up sẽ bị vô hiệu hóa trong suốt quá trình Reset. Nếu sự tổn hao nguồn thấp trong suốt quá trình Reset là quan trọng , nó được khuyến cáo để sử dụng 1 pull-up ngoài hoặc 1 pull-down . Việc kết nối các chân không sử dụng một cách trực tiếp đến  $V_{CC}$  và GND thì không được khuyến khích , từ khi việc này gây ra dòng điện thừa nếu chân này được cấu hình ngẫu nhiên như là 1 đầu ra .

### Chức năng cổng luân phiên

Tất cả các chân cổng đều có chức năng luân phiên bổ sung vào để trở thành các I/O số chung . Hình 33 chỉ ra cách 1 chân cổng điều khiển các tín hiệu rút gọn từ hình 30 có thể được ghi đè bằng các cổng chức năng luân phiên (alternate Functions) . Sự ghi đè tín hiệu có thể không được đưa ra trong tất cả các chân cổng , nhưng hình này phục vụ như là một sự miêu tả chung có thể áp dụng lên tất cả các chân cổng trong họ vi điều khiển AVR

**Figure 33. Alternate Port Functions<sup>(1)</sup>**

PUDExn:	Pxn PULLUP OVERRIDE ENABLE	PUD:	PULLUP DISABLE
PUDVxn:	Pxn PULLUP OVERRIDE VALUE	WDx:	WRITE DDX <sub>x</sub>
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE	RDx:	READ DDX <sub>x</sub>
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RPx:	READ PORT <sub>x</sub> REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WPx:	WRITE PORT <sub>x</sub>
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	PPx:	READ PORT <sub>x</sub> PIN
DIOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDEENABLE	dk <sub>IO</sub> :	IO CLOCK
DIOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE	Dxx:	DIGITAL INPUT PIN n ON PORT <sub>x</sub>
SLEEP:	SLEEP CONTROL	AIxnx:	ANALOG INPUT/OUTPUT PIN n ON PORT <sub>x</sub>

Note: 1. WP<sub>x</sub>, WD<sub>x</sub>, RL<sub>x</sub>, RP<sub>x</sub>, and RD<sub>x</sub> are common to all pins within the same port. clk<sub>IO</sub>, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

Bảng 26 tổng kết chức năng của các tín hiệu ghi đè (Overriding signals). Các danh mục các chân và cổng từ hình 33 thì không được chỉ ra trong các bảng kế tiếp. Các tín hiệu ghi đè thì được sinh ra bên trong các module có chức năng luân phiên (alternate Function).

**Table 26. Generic Description of Overriding Signals for Alternate Functions.**

Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD Register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn Register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn Register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn Register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn Register bit.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU-state (Normal mode, Sleep modes).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, Sleep modes).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the schmitt trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/Output	This is the Analog Input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

Vùng nhỏ bên dưới miêu tả chức năng luân phiên cho mỗi cổng , và có liên quan đến các tín hiệu ghi đè lên các chức năng luân phiên . Tham khảo thêm phần mô tả chức năng luân phiên để biết thêm chi tiết

### Thanh ghi IO chử năng đặc biệt – SFIOR

Bit	7	6	5	4	3	2	1	0	SFIOR
Read/Write	TSM	-	-	-	ACME	PUD	PSR0	PSR321	
Initial Value	R/W	R	R	R	R/W	R/W	R/W	R/W	

Bit 2 – PUD : vô hiệu hóa pull-up

Khi bit này được viết là 1 , pull-up trong các cổng I/O bị vô hiệu hóa dù là thanh ghi DDxn và PORTxn được cấu hình để kích hoạt pull-ups ({DDxn , PORTxn }=0b01 ). Xem phần cấu hình các chân ở trang 67 để biết thêm chi tiết về đặc điểm này .

### Chức năng luân phiên của cổng A

Cổng A có 1 hàm luân phiên Alternate Function như là địa chỉ byte thấp và dòng dữ liệu cho giao diện bộ nhớ ngoài .

**Table 27.** Port A Pins Alternate Functions

Port Pin	Alternate Function
PA7	AD7 (External memory interface address and data bit 7)
PA6	AD6 (External memory interface address and data bit 6)
PA5	AD5 (External memory interface address and data bit 5)
PA4	AD4 (External memory interface address and data bit 4)
PA3	AD3 (External memory interface address and data bit 3)
PA2	AD2 (External memory interface address and data bit 2)
PA1	AD1 (External memory interface address and data bit 1)
PA0	AD0 (External memory interface address and data bit 0)

Bảng 28 và 29 liên quan đến hàm luân phiên của cổng A đến các tín hiệu ghi đè được chỉ ra trong hình 33 trang 71.

**Table 28.** Overriding Signals for Alternate Functions in PA7..PA4

Signal Name	PA7/AD7	PA6/AD6	PA5/AD5	PA4/AD4
PUOE	SRE	SRE	SRE	SRE
PUOV	$\sim(\overline{WR} \mid ADA^{(1)}) \cdot PORTA7 \cdot PUD$	$\sim(\overline{WR} \mid ADA) \cdot PORTA6 \cdot PUD$	$\sim(\overline{WR} \mid ADA) \cdot PORTA5 \cdot PUD$	$\sim(\overline{WR} \mid ADA) \cdot PORTA4 \cdot PUD$
DDOE	SRE	SRE	SRE	SRE
DDOV	$\overline{WR} \mid ADA$	$\overline{WR} \mid ADA$	$\overline{WR} \mid ADA$	$\overline{WR} \mid ADA$
PVOE	SRE	SRE	SRE	SRE
PVOV	$A7 \cdot ADA \mid D7$ OUTPUT • WR	$A6 \cdot ADA \mid D6$ OUTPUT • WR	$A5 \cdot ADA \mid D5$ OUTPUT • WR	$A4 \cdot ADA \mid D4$ OUTPUT • WR
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	D7 INPUT	D6 INPUT	D5 INPUT	D4 INPUT
AI0	-	-	-	-

Note: 1. ADA is short for ADdress Active and represents the time when address is output. See "External Memory Interface" on page 26 for details.

**Table 29.** Overriding Signals for Alternate Functions in PA3..PA0

Signal Name	PA3/AD3	PA2/AD2	PA1/AD1	PA0/AD0
PUOE	SRE	SRE	SRE	SRE
PUOV	$\sim(\overline{WR} \mid ADA) \cdot PORTA3 \cdot PUD$	$\sim(\overline{WR} \mid ADA) \cdot PORTA2 \cdot PUD$	$\sim(\overline{WR} \mid ADA) \cdot PORTA1 \cdot PUD$	$\sim(\overline{WR} \mid ADA) \cdot PORTA0 \cdot PUD$
DDOE	SRE	SRE	SRE	SRE
DDOV	$\overline{WR} \mid ADA$	$\overline{WR} \mid ADA$	$\overline{WR} \mid ADA$	$\overline{WR} \mid ADA$
PVOE	SRE	SRE	SRE	SRE
PVOV	$A3 \cdot ADA \mid D3$ OUTPUT • WR	$A2 \cdot ADA \mid D2$ OUTPUT • WR	$A1 \cdot ADA \mid D1$ OUTPUT • WR	$A0 \cdot ADA \mid D0$ OUTPUT • WR
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	D3 INPUT	D2 INPUT	D1 INPUT	D0 INPUT
AI0	-	-	-	-

## Chức năng luân phiên của cổng B : Alternate Functions of Port B

**Table 30.** Port B Pins Alternate Functions

Port Pin	Alternate Functions
PB7	OC2/OC1C <sup>(1)</sup> (Output Compare and PWM Output for Timer/Counter2 or Output Compare and PWM Output C for Timer/Counter1)
PB6	OC1B (Output Compare and PWM Output B for Timer/Counter1)
PB5	OC1A (Output Compare and PWM Output A for Timer/Counter1)
PB4	OC0 (Output Compare and PWM Output for Timer/Counter0)
PB3	MISO (SPI Bus Master Input/Slave Output)
PB2	MOSI (SPI Bus Master Output/Slave Input)
PB1	SCK (SPI Bus Serial Clock)
PB0	SS (SPI Slave Select input)

Note: 1. OC1C not applicable in ATmega103 compatibility mode.

Chân luân phiên được cấu hình như bên dưới :

OC2/OC1C , bit 7

OC2 , đầu ra so sánh ghép với đầu ra : chân PB7 có thể phục vụ như là một đầu ra bên ngoài cho đầu ra so sánh Timer/Counter 2. Chân này được cấu hình như là một cổng ra (DDB7 đặt là 1 ) để phục vụ chức năng này . Chân OC2 cũng là đầu ra cho chức năng timer mode của PWM .

OC1C , đầu ra ghép với đầu ra C . Chân PB7 có thể phục vụ như là một đầu ra bên ngoài cho đầu ra so sánh C của Timer/Counter1 . Chân này có thể được cấu hình như là 1 đầu ra (DDB7 được đặt là 1 )để phục vụ chức năng này . Chân OC1C cũng là đầu ra cho chức năng timer PWM .

OC1B , bit 6

OC1B , đầu ra so sánh ghép đầu ra B : Chân PB6 có thể phục vụ như là 1 đầu ra bên ngoài cho đầu ra so sánh B của Timer/Counter 1 . Chân này phải được cấu hình như là 1 đầu ra (DDB6 đặt là 1 ) để phục vụ chức năng này . Chân OC1B cũng là chân ra cho chức năng PWM mode timer

OC1A , bit 5

OC1A , đầu ra so sánh ghép đầu ra A : chân PB5 có thể phục vụ như là một cổng ra bên ngoài cho Timer/Counter 1 đầu ra so sánh A . Chân này phải được cấu hình như là 1 cổng ra ( DDB5 đặt là 1 ) để phục vụ chức năng này . Chân OC1A cũng là chân đầu ra cho chức năng PWM mode timer .

OC0 , Bit 4

OC0 , đầu ra so sánh ghép với đầu ra : Chân PB4 có thể phục vụ như là 1 đầu ra bên ngoài cho đầu ra so sánh Timer/Counter 0 . Chân này phải được cấu hình như là 1 đầu ra (DDB4 đặt là 1 ) để phục vụ cho chức năng này . Chân OC0 cũng là đầu ra cho chức năng PWM mode timer.

MISO – cổng B , bit 3

MISO : chân Master Data input , Slave Data output cho kênh SPI . Khi mà SPI được kích hoạt như là một master , chân này được cấu hình như là 1 đầu ra mà không quan tâm đến việc cài đặt của DDB3 . Khi SPI được kích hoạt như là 1 slave , sự định hướng dữ liệu của chân này được điều khiển bởi DDB3 . Khi mà chân này bị ép buộc làm 1 đầu vào , pull-up có thể vẫn được điều khiển bằng bit PORTB3

#### MOSI – Cổng B , bit 2

MOSI : SPI Master Data ouput , Slave Data input cho kênh SPI . Khi mà SPI được kích hoạt như là slave , chân này được cấu hình như là một đầu vào bắt chấp sự cài đặt của DDB2 . Khi SPI được cấu hình như 1 master , sự định hướng dữ liệu của chân này được điều khiển bằng DDB2 . Khi chân này bị ép buộc làm 1 đầu vào , pull-up có thể vẫn được điều khiển bởi bit PORTB2 .

#### SCK – cổng B , bit 1

SCK : Master Clock output , Slave Clock input cho kênh SPI . Khi mà SPI được kích hoạt như là slave , chân này được cấu hình như là một đầu vào bắt chấp việc cài đặt của DDB1. Khi SPI được kích hoạt như là một master , sự định hướng dữ liệu của chân này được điều khiển bằng DDB1. Khi chân này bị ép buộc làm 1 đầu vào , pull-up có thể vẫn được điều khiển bằng bit PORTB1

#### SS – cổng B ,bit 0

SS : Slave Port Select input . Khi mà SPI được kích hoạt như là 1 slave , chân này được cấu hình như là 1 đầu vào bắt chấp việc cài đặt của DDB0 . Như là 1 slave , SPI được kích hoạt khi chân này bị động ở mức thấp . Khi SPI được kích hoạt như là một master , sự định hướng dữ liệu của chân này được điều khiển bởi DDB0 . Khi chân này bị ép buộc làm một đầu vào, pull-up có thể vẫn được điều khiển bằng bit PORTB0 Bảng 31 và 32 có liên quan đến chức năng luân phiên của cổng B lên các tín hiệu ghi đè được chỉ ra trong bảng 33 trang 71 . SPI MSTR INPUT và SPI SLAVE OUTPUT cấu thành tín hiệu MISO , trong khi MOSI bị chia ra trong SPI MSTR OUTPUT và SPI SLAVE INPUT

**Table 31.** Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/OC2/OC1C	PB6/OC1B	PB5/OC1A	PB4/OC0
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	OC2/OC1C ENABLE <sup>(1)</sup>	OC1B ENABLE	OC1A ENABLE	OC0 ENABLE
PVOV	OC2/OC1C <sup>(1)</sup>	OC1B	OC1A	OC0B
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	-	-	-	-
AI0	-	-	-	-

Note: 1. See "Output Compare Modulator (OCM1C2)" on page 161 for details. OC1C does not exist in ATmega103 compatibility mode.

**Table 32.** Overriding Signals for Alternate Functions in PB3..PB0

Signal Name	PB3/MISO	PB2/MOSI	PB1/SCK	PB0/SS
PUOE	SPE • MSTR	SPE • MSTR	SPE • MSTR	SPE • MSTR
PUOV	PORTB3 • PUD	PORTB2 • PUD	PORTB1 • PUD	PORTB0 • PUD
DDOE	SPE • MSTR	SPE • MSTR	SPE • MSTR	SPE • MSTR
DDOV	0	0	0	0
PVOE	SPE • MSTR	SPE • MSTR	SPE • MSTR	0
PVOV	SPI SLAVE OUTPUT	SPI MSTR OUTPUT	SCK OUTPUT	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	SPI MSTR INPUT	SPI SLAVE INPUT	SCK INPUT	SPI SS
AI0	-	-	-	-

## Chức năng luân phiên của cổng C

Trong chế độ tương thích với Atmega 103 , cổng C chỉ là cổng ra . Atmega 128 thì vận chuyển mặc định ( default shipped ) trong chế độ tương thích . Vì vậy , nếu các phần không được lập trình trước khi chúng được đặt trên PCB , PORTC sẽ là cổng ra trong suốt quá trình bật nguồn đầu tiên , và cho đến khi chế độ tương thích với Atmega 103 bị vô hiệu hóa . Cổng C có hàm chức năng luân phiên (alternate Function ) như là các byte địa chỉ cao cho giao diện bộ nhớ ngoài

**Table 33.** Port C Pins Alternate Functions

Port Pin	Alternate Function
PC7	A15
PC6	A14
PC5	A13
PC4	A12
PC3	A11
PC2	A10
PC1	A9
PC0	A8

Bảng 34 và 35 liên quan đến chức năng luân phiên của cổng C đến việc ghi đè tín hiệu được chỉ ra trong hình 33 ở trang 71

**Table 34.** Overriding Signals for Alternate Functions in PC7..PC4

Signal Name	PC7/A15	PC6/A14	PC5/A13	PC4/A12
PUOE	SRE • (XMM <sup>(1)</sup> <1)	SRE • (XMM<2)	SRE • (XMM<3)	SRE • (XMM<4)
PUOV	0	0	0	0
DDOE	SRE • (XMM<1)	SRE • (XMM<2)	SRE • (XMM<3)	SRE • (XMM<4)
DDOV	1	1	1	1
PVOE	SRE • (XMM<1)	SRE • (XMM<2)	SRE • (XMM<3)	SRE • (XMM<4)
PVOV	A15	A14	A13	A12
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AI0	–	–	–	–

Note: 1. XMM = 0 in ATmega103 compatibility mode.

**Table 35.** Overriding Signals for Alternate Functions in PC3..PC0<sup>(1)</sup>

Signal Name	PC3/A11	PC2/A10	PC1/A9	PC0/A8
PUOE	SRE • (XMM<5)	SRE • (XMM<6)	SRE • (XMM<7)	SRE • (XMM<7)
PUOV	0	0	0	0
DDOE	SRE • (XMM<5)	SRE • (XMM<6)	SRE • (XMM<7)	SRE • (XMM<7)
DDOV	1	1	1	1
PVOE	SRE • (XMM<5)	SRE • (XMM<6)	SRE • (XMM<7)	SRE • (XMM<7)
PVOV	A11	A10	A9	A8
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AI0	–	–	–	–

Note: 1. XMM = 0 in ATmega103 compatibility mode.

## Chức năng luân phiên của cổng D (alternate Functions of Port D )

Các chân cổng D với các chức năng luân phiên được chỉ ra trong bảng 36

**Table 36. Port D Pins Alternate Functions**

Port Pin	Alternate Function
PD7	T2 (Timer/Counter2 Clock Input)
PD6	T1 (Timer/Counter1 Clock Input)
PD5	XCK1 <sup>(1)</sup> (USART1 External Clock Input/Output)
PD4	ICP1 (Timer/Counter1 Input Capture Pin)
PD3	INT3/TXD1 <sup>(1)</sup> (External Interrupt3 Input or UART1 Transmit Pin)
PD2	INT2/RXD1 <sup>(1)</sup> (External Interrupt2 Input or UART1 Receive Pin)
PD1	INT1/SDA <sup>(1)</sup> (External Interrupt1 Input or TWI Serial DAta)
PD0	INT0/SCL <sup>(1)</sup> (External Interrupt0 Input or TWI Serial CLock)

Note: 1. XCK1, TXD1, RXD1, SDA, and SCL not applicable in ATmega103 compatibility mode.

Cấu hình các chân luân phiên thì được chỉ ra dưới đây :

T2 – cổng D , bit 7

T2 , nguồn bộ đếm Timer/Counter 2

T1 – Cổng D , bit 6

T1, nguồn bộ đếm Timer/Counter 1

XCK1 – Cổng D ,bit 5

XCK1 , xung nhịp ngoài USART1 . Thanh ghi định hướng dữ liệu (DDD4) điều khiển 1 trong hai xung nhịp là đầu ra (DDD4 được đặt ) hoặc là đầu vào (DDD4 bị xóa ). Chân XCK1 thì hoạt động chỉ khi USART1 hoạt động trong chế độ đồng bộ hóa .

ICP1 – cổng D ,bit 4

ICP1 – Input Capture Pin1 : Chân PD4 có thể đóng vai trò như là một chân thu thập đầu vào cho Timer/Counter 1

INT3/TXD1 – Cổng D , bit 3

INT3 , External Interrupt source 3 : Chân PD3 có thể phục vụ như là 1 nguồn ngắt ngoài đến MCU

TXD1 , Transmit Data (chân đầu ra dữ liệu cho USART1 ) . Khi mà bộ chuyển phát USART1 được kích hoạt , chân này được cấu hình như là một đầu ra bắt chấp giá trị của DDD3

INT2/RXD1 – cổng D , bit 2

INT2 , External Interrupt source 2 . Chân PD2 có thể phục vụ như là 1 nguồn ngắt ngoài đến MCU

RXD1 , Receive Data ( chân đầu vào dữ liệu cho USART1 ) . Khi mà bộ thu USART1 được kích hoạt , chân này được cấu hình như là 1 đầu vào bắt chấp giá trị của DDD2. Khi USART bắt ép chân này trở thành 1 đầu vào , pull-up có thể vẫn được điều khiển bởi bit PORTD2

INT1/SDA – cổng D , bit 1

INT1 , External Interrupt source 1 . Chân PD1 có thể phục vụ như là 1 nguồn ngắt ngoài đến MCU

SDA , Two-wire Serial Interface Data : khi mà bit TWEN trong thanh ghi TWCR được đặt là 1 để kích hoạt giao diện 2 dây nối tiếp , chân PD1 thì bị ngắt kết nối khỏi cổng và trở thành chân I/O dữ liệu nối tiếp cho giao diện nối tiếp 2 dây . Trong chế độ này , có 1 bộ lọc xung nhiễu trên chân để khử các tín hiệu ngắn hơn 50ns trên tín hiệu đầu vào , và chân này được điều khiển bởi 1 bộ điều khiển kênh mở với sự giới hạn tốc độ quay .

INT0/SCL – Cổng D , bit 0

INT0 , External Interrupt source 0 . Chân PD0 có thể phục vụ như là 1 nguồn ngắt bên ngoài đến MCU

SCL , Two-wire Serial Interface Clock : Khi mà TWEN trong TWCR được đặt là 1 để kích hoạt giao diện 2 dây nối tiếp , chân PD0 bị ngắt kết nối khỏi cổng và trở thành cổng I/O xung nhịp nối tiếp cho giao diện 2 dây nối tiếp . Trong chế độ này , có một bộ lọc xung nhiễu trên chân để khử các xung nhiễu ngắn hơn 50ns trên tín hiệu đầu vào và chân này được điều khiển bởi 1 bộ điều khiển kênh mở với sự giới hạn tốc độ quay .

Bảng 37 và 38 liên quan đến chức năng luân phiên của cổng D đến tín hiệu ghi đè được chỉ ra trong bảng 33 trang 71

**Table 37.** Overriding Signals for Alternate Functions PD7..PD4

Signal Name	PD7/T2	PD6/T1	PD5/XCK1	PD4/ICP1
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	UMSEL1	0
PVOV	0	0	XCK1 OUTPUT	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	T2 INPUT	T1 INPUT	XCK1 INPUT	ICP1 INPUT
AIO	-	-	-	-

**Table 38.** Overriding Signals for Alternate Functions in PD3..PD0<sup>(1)</sup>

Signal Name	PD3/INT3/TXD1	PD2/INT2/RXD1	PD1/INT1/SDA	PD0/INT0/SCL
PUOE	TXEN1	RXEN1	TWEN	TWEN
PUOV	0	PORTD2 • PUD	PORTD1 • PUD	PORTD0 • PUD
DDOE	TXEN1	RXEN1	TWEN	TWEN
DDOV	1	0	SDA_OUT	SCL_OUT
PVOE	TXEN1	0	TWEN	TWEN
PVOV	TXD1	0	0	0
DIEOE	INT3 ENABLE	INT2 ENABLE	INT1 ENABLE	INT0 ENABLE
DIEOV	1	1	1	1
DI	INT3 INPUT	INT2 INPUT/RXD1	INT1 INPUT	INT0 INPUT
AIO	-	-	SDA INPUT	SCL INPUT

Note: 1. When enabled, the Two-wire Serial Interface enables Slew-Rate controls on the output pins PD0 and PD1. This is not shown in this table. In addition, spike filters are connected between the AIO outputs shown in the port figure and the digital logic of the TWI module.

## Chức năng luân phiên của cổng E ( Alternate Functions of Port E )

Các chân cổng E với chức năng luân phiên được chỉ ra trong bảng 39

**Table 39.** Port E Pins Alternate Functions

Port Pin	Alternate Function
PE7	INT7/ICP3 <sup>(1)</sup> (External Interrupt 7 Input or Timer/Counter3 Input Capture Pin)
PE6	INT6/ T3 <sup>(1)</sup> (External Interrupt 6 Input or Timer/Counter3 Clock Input)
PE5	INT5/OC3C <sup>(1)</sup> (External Interrupt 5 Input or Output Compare and PWM Output C for Timer/Counter3)
PE4	INT4/OC3B <sup>(1)</sup> (External Interrupt4 Input or Output Compare and PWM Output B for Timer/Counter3)
PE3	AIN1/OC3A <sup>(1)</sup> (Analog Comparator Negative Input or Output Compare and PWM Output A for Timer/Counter3)
PE2	AIN0/XCK0 <sup>(1)</sup> (Analog Comparator Positive Input or USART0 external clock input/output)
PE1	PDO/TXD0 (Programming Data Output or UART0 Transmit Pin)
PE0	PDI/RXD0 (Programming Data Input or UART0 Receive Pin)

Note: 1. ICP3, T3, OC3C, OC3B, OC3A, and XCK0 not applicable in ATmega103 compatibility mode.

## INT7/ICP3 – cổng E , bit 7

INT7 , External Interrupt source 7 : Chân PE7 có thể phục vụ như là một nguồn ngắt ngoài.

ICP3 – Input Capture Pin 3 : Chân PE7 có thể đóng vai trò như là một chân thu thập đầu vào cho Timer/Counter 3

## INT6/T3 – Cổng E ,bit 6

INT6 , External Interrupt source : chân PE6 có thể phục vụ như là 1 nguồn ngắt ngoài

T3 , nguồn bộ đếm Timer/Counter 3

## INT5/OC3C – Cổng E , bit 5

INT5 , External Interrupt source 5 : Chân PE5 có thể phục vụ như là một nguồn ngắt ngoài

OC3C , Output Compare Match C ouput ( đầu ra so sánh tương ứng đầu ra C ) : chân PE5 có thể phục vụ như là một đầu ra bên ngoài cho đầu ra so sánh cổng C Timer/Counter 3 . Chân này phải được cấu hình như là một đầu ra (DDE5 đặt là 1 ) để phục vụ chức năng này . Chân OC3C cũng là chân đầu ra cho chức năng Timer trong chế độ PWM

## INT4/OC3B – Cổng E ,bit 4

INT4 , External Interrupt source 4 : chân PE4 có thể phục vụ như là 1 nguồn ngắt ngoài

OC3B , đầu ra so sánh ghép đầu ra B : Chân PE4 có thể phục vụ như là một đầu ra bên ngoài cho đầu ra so sánh B của Timer/Counter 3 . Chân này phải được cấu hình như là một cổng ra (DDE4 đặt là 1 ) để phục vụ chức năng này . Chân OC3B cũng là chân đầu ra cho chức năng timer trong chế độ PWM .

## AIN1/OC3A – cổng E ,bit 3

AIN1 – Analog Comparator Negative Input . Chân này được nối trực tiếp đến đầu vào âm của bộ so sánh tương tự (analog Comparator )

OC3A , đầu ra so sánh ghép đầu ra A : Chân PE3 có thể phục vụ như là 1 đầu ra bên ngoài cho đầu ra so sánh A của Timer/Counter 3 . Chân này được cấu hình như là 1 đầu ra (DDE3 đặt là 1 )để phục vụ chức năng này . Chân OC3A cũng là chân ra cho chức năng timer trong chế độ PWM

## AIN0/XCK0 – cổng E , bit 2

AIN0 – Analog Comparator Positive input (đầu vào dương của bộ so sánh Analog ) : chân này được nối trực tiếp với đầu vào dương của bộ so sánh analog

XCK0 , USART0 External Clock ( xung nhịp ngoài của USART0) . Thanh ghi định hướng dữ liệu (DDE2) điều khiển 1 trong 2 xung nhịp là đầu ra (DDE2 được đặt ) hoặc là đầu vào (DDE2 bị xóa ) . Chân XCK0 chỉ hoạt động khi USART0 điều khiển trong chế độ đồng bộ hóa

## PDO/TXD0 – Cổng E ,bit 1

PDO , SPI Serial Programming Data Output (đầu ra dữ liệu lập trình nối tiếp SPI) . Trong suốt quá trình tải về chương trình nối tiếp , chân này được sử dụng như là 1 dòng đầu vào dữ liệu cho Atmega 128

RXD0 , USART0 Receive Pin ( chân dữ liệu đầu vào đến USART0) . Khi bộ thu tín hiệu USART0 được kích hoạt chân này được cấu hình là 1 như là một đầu vào bất chấp giá trị của DDRE0 . Khi USART0 bắt chân này là một đầu vào , 1 mức logic 1 trong PORTE0 sẽ bật pull-up bên trong lên

Bảng 40 và 41 liên quan đến chức năng luân phiên của cổng E tới việc ghi đè tín hiệu được chỉ ra trong hình 33 trang 71

**Table 40.** Overriding Signals for Alternate Functions PE7..PE4

Signal Name	PE7/INT7/ICP3	PE6/INT6/T3	PE5/INT5/OC3C	PE4/INT4/OC3B
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	OC3C ENABLE	OC3B ENABLE
PVOV	0	0	OC3C	OC3B
DIEOE	INT7 ENABLE	INT6 ENABLE	INT5 ENABLE	INT4 ENABLE
DIEOV	1	1	1	1
DI	INT7 INPUT/ICP3 INPUT	INT7 INPUT/T3 INPUT	INT5 INPUT	INT4 INPUT
AIO	-	-	-	-

**Table 41.** Overriding Signals for Alternate Functions in PE3..PE0

Signal Name	PE3/AIN1/OC3A	PE2/AIN0/XCK0	PE1/PDO/TXDO	PE0/PDI/RXD0
PUOE	0	0	TXENO	RXENO
PUOV	0	0	0	PORTE0 • PUD
DDOE	0	0	TXENO	RXENO
DDOV	0	0	1	0
PVOE	OC3B ENABLE	UMSEL0	TXENO	0
PVOV	OC3B	XCK0 OUTPUT	TXDO	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	0	XCK0 INPUT	-	RXD0
AIO	AIN1 INPUT	AIN0 INPUT	-	-

### Chức năng luân phiên của cổng F ( alternate Functions of port F )

Cổng F có một chức năng luân phiên như là đầu vào tương tự cho bộ chuyển đổi ADC như là được chỉ ra trong bảng 42 . Nếu một vài chân của cổng F được cấu hình như là những đầu ra , điều này là cần thiết vì không có bộ chuyển mạch khi 1 quá trình chuyển đổi đang được tiến hành . Điều này có thể làm hỏng kết quả của quá trình chuyển đổi . Trong chế độ tương thích với Atmega 103 thì chỉ cổng F được sử dụng. Nếu như giao diện JTAG được kích hoạt , điện trở pull-up trên chân PF7 (TDI) , PF5(TMS) và PF4(TCK) sẽ được kích hoạt trừ khi tín hiệu Reset xuất hiện

**Table 42. Port F Pins Alternate Functions**

Port Pin	Alternate Function
PF7	ADC7/TDI (ADC input channel 7 or JTAG Test Data Input)
PF6	ADC6/TDO (ADC input channel 6 or JTAG Test Data Output)
PF5	ADC5/TMS (ADC input channel 5 or JTAG Test Mode Select)
PF4	ADC4/TCK (ADC input channel 4 or JTAG Test Clock)
PF3	ADC3 (ADC input channel 3)
PF2	ADC2 (ADC input channel 2)
PF1	ADC1 (ADC input channel 1)
PF0	ADC0 (ADC input channel 0)

TDI, ACD7 – cổng F , bit 7

ACD7 , bộ chuyển đổi tương tự sang số , kênh 7

TDI , JTAG Test Data In : dữ liệu cổng vào nối tiếp được shifted trong thanh ghi lệnh của thành chi dữ liệu ( các chuỗi quét ). Khi mà giao diện JTAG được kích hoạt , chân này có thể không được sử dụng như là một chân I/O

TDO,ADC6 – cổng F ,bit 6

ADC6 , bộ chuyển đổi tương tự sang số , kênh 6

TDO, JTAG Test Data Out : dữ liệu đầu vào nối tiếp từ thanh ghi lệnh hoặc thanh ghi dữ liệu . Khi giao diện JTAG được kích hoạt, chân này không thể được sử dụng như là một chân I/O

Chân TDO có 3 trạng thái trừ phi các trạng thái TAP shift dữ liệu ngoài được truy nhập

TMS, ADC5 – cổng F , bit 5

ADC5 , bộ chuyển đổi tương tự sang số , kênh 5

TMS , lựa chọn chế độ kiểm tra JTAG : chân này được sử dụng cho sự định hướng thông qua bộ điều khiển TAP trạng thái máy (TAP-controller state machine ). Khi giao diện JTAG được kích hoạt chân này không thể được sử dụng như là 1 chân I/O

TCK, ACD4 – cổng F ,bit 4

ACD4 , bộ chuyển đổi tương tự sang số ,kênh 4

TCK , xung nhịp kiểm tra JTAG : quá trình điều khiển JTAG là đồng bộ hóa lên TCK . Khi mà giao diện JTAG được kích hoạt , chân này không thể sử dụng như là một chân I/O

ADC3 – ADC0 – cổng F , bit 3..0

Bộ chuyển đổi tương tự sang số , kênh 3...0

**Table 43.** Overriding Signals for Alternate Functions in PF7..PF4

Signal Name	PF7/ADC7/TDI	PF6/ADC6/TDO	PF5/ADC5/TMS	PF4/ADC4/TCK
PUOE	JTAGEN	JTAGEN	JTAGEN	JTAGEN
PUOV	1	0	1	1
DDOE	JTAGEN	JTAGEN	JTAGEN	JTAGEN
DDOV	0	SHIFT_IR + SHIFT_DR	0	0
PVOE	0	JTAGEN	0	0
PVOV	0	TDO	0	0
DIEOE	JTAGEN	JTAGEN	JTAGEN	JTAGEN
DIEOV	0	0	0	0
DI	-	-	-	-
AIO	TDI/ADC7 INPUT	ADC6 INPUT	TMS/ADC5 INPUT	TCK/ADC4 INPUT

**Table 44.** Overriding Signals for Alternate Functions in PF3..PF0

Signal Name	PF3/ADC3	PF2/ADC2	PF1/ADC1	PF0/ADC0
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	-	-	-	-
AIO	ADC3 INPUT	ADC2 INPUT	ADC1 INPUT	ADC0 INPUT

## Chức năng luân phiên của cổng G

Trong chế độ tương thích với Atmega 103 , chỉ chế độ luân phiên là mặc định cho cổng G , và cổng G không thể được sử dụng như là các chân cổng số chung . Cấu hình các chân luân phiên như là được bên dưới :

**Table 45.** Port G Pins Alternate Functions

Port Pin	Alternate Function
PG4	TOSC1 (RTC Oscillator Timer/Counter0)
PG3	TOSC2 (RTC Oscillator Timer/Counter0)
PG2	ALE (Address Latch Enable to external memory)
PG1	RD (Read strobe to external memory)
PG0	WR (Write strobe to external memory)

### TOSC1 – cổng G , bit 4

TOSC1 , chân bộ tạo dao động Timer : khi mà bit AS0 trong thanh ghi ASSR được đặt là 1 để kích hoạt bộ định thời dị bộ của Timer/Counter 0 , chân PG3 bị ngắt kết nối khỏi các cổng , và trở thành đầu ra cho bộ khuỷch đại của bộ khuỷch đại dao động ké (Oscillator amplifier ) . Trong chế độ này , 1 bộ tạo dao động thạch anh được kết nối với các chân này , và các chân này không được sử dụng như là 1 chân vào ra I/O

### TOSC2 – cổng G , bit 3

TOSC2 , chân 2 của bộ tạo dao động của Timer : Khi bit AS0 trong thanh ghi ASSR được đặt là 1 để kích hoạt bộ định thời dị bộ của Timer/Counter 0 , chân PG3 bị ngắt kết nối khỏi cổng , và trở thành đầu ra khuyêch đại của bộ khuyêch đại của bộ tạo dao động . Trong chế độ này , 1 bộ tạo dao động thạch anh được kết nối với chân này ,và chân này không thể được sử dụng như là một chân vào ra .

### ALE – cổng G , bit 2

ALE là bộ nhớ dữ liệu bên ngoài Address Latch Enable signal

### RD – cổng G , bit 1

RD là bộ nhớ dữ liệu bên ngoài đọc điều khiển phân tích

### WR – cổng G , bit 0

WR là bộ nhớ dữ liệu bên ngoài viết điều khiển phân tích

Bảng 46 và 47 liên quan đến chức năng luân phiên của cổng G tới việc ghi đè tín hiệu được chỉ ra trong hình 33 trang 71

**Table 46.** Overriding Signals for Alternate Functions in PG4..PG1

Signal Name	PG4/TOSC1	PG3/TOSC2	PG2/ALE	PG1/RD
PUOE	AS0	AS0	SRE	SRE
PUOV	0	0	0	0
DDOE	AS0	AS0	SRE	SRE
DDOV	0	0	1	1
PVOE	0	0	SRE	SRE
PVOV	0	0	ALE	RD
DIEOE	AS0	AS0	0	0
DIEOV	0	0	0	0
DI	-	-	-	-
AIO	T/C0 OSC INPUT	T/C0 OSC OUTPUT	-	-

**Table 47.** Overriding Signals for Alternate Functions in PG0

Signal Name	PG0/WR
PUOE	SRE
PUOV	0
DDOE	SRE
DDOV	1
PVOE	SRE
PVOV	WR
DIEOE	0
DIEOV	0
DI	-
AIO	-

## Sự mô tả thanh ghi cho các cổng I/O

Thanh ghi dữ liệu cổng A - PORTA

Bit	7	6	5	4	3	2	1	0	
	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cổng A – DDRA

Bit	7	6	5	4	3	2	1	0	
	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân đầu vào cổng A - PINA

Bit	7	6	5	4	3	2	1	0	
	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
ReadWrite	R	R	R	R	R	R	R	R	
Initial Value	N/A								

Thanh ghi dữ liệu cổng B – PORTB

Bit	7	6	5	4	3	2	1	0	
	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cổng B – DDRB

Bit	7	6	5	4	3	2	1	0	
	DDR7	DDR6	DDR5	DDR4	DDR3	DDR2	DDR1	DDR0	DDRB
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân đầu vào B – PINB

Bit	7	6	5	4	3	2	1	0	
	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
ReadWrite	R	R	R	R	R	R	R	R	
Initial Value	N/A								

Thanh ghi dữ liệu cổng C – PORTC

Bit	7	6	5	4	3	2	1	0	
	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	PORTC
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cổng C – DDRC

Bit	7	6	5	4	3	2	1	0	
ReadWrite	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân đầu vào C – PINC

Bit	7	6	5	4	3	2	1	0	
ReadWrite	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
Initial Value	N/A								

Trong chế độ tương thích với Atmega 103 , thanh ghi DDRC và PINC được khởi tạo để đẩy và kéo ( Push – Pull )đầu ra Zero . Các chân cổng giả sử ở giá trị khởi tạo của chúng , cho dù bộ định thời đang không hoạt động . Chú ý rằng các thanh ghi DDRC và PINC thì không khả dụng trong chế độ tương thích với Atmega 103 và nên không được sử dụng cho chế độ tương thích phía sau .

Thanh ghi dữ liệu cổng D – PORTD

Bit	7	6	5	4	3	2	1	0	
ReadWrite	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cho cổng D – DDRD

Bit	7	6	5	4	3	2	1	0	
ReadWrite	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân đầu vào của cổng D

Bit	7	6	5	4	3	2	1	0	
ReadWrite	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Initial Value	N/A								

Thanh ghi dữ liệu cho cổng E – PORTE

Bit	7	6	5	4	3	2	1	0	
ReadWrite	PORTE7	PORTE6	PORTE5	PORTE4	PORTE3	PORTE2	PORTE1	PORTE0	PORTE
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cho cổng E – DDRE

Bit	7	6	5	4	3	2	1	0	
ReadWrite	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0	DDRE
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân cổng đầu vào cổng E – PINE

Bit	7	6	5	4	3	2	1	0	
	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0	PINF
ReadWrite	R	R	R	R	R	R	R	R	
Initial Value	N/A								

Thanh ghi dữ liệu cổng F – PORTF

Bit	7	6	5	4	3	2	1	0	
	PORTF7	PORTF6	PORTF5	PORTF4	PORTF3	PORTF2	PORTF1	PORTF0	PORTF
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cổng F – DDRF

Bit	7	6	5	4	3	2	1	0	
	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0	DDRF
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân đầu vào – PINF

Bit	7	6	5	4	3	2	1	0	
	PINF7	PINF6	PINF5	PINF4	PINF3	PINF2	PINF1	PINF0	PINF
ReadWrite	R	R	R	R	R	R	R	R	
Initial Value	N/A								

Thanh ghi dữ liệu cổng G – PORTG

Bit	7	6	5	4	3	2	1	0	
	-	-	-	PORTG4	PORTG3	PORTG2	PORTG1	PORTG0	PORTG
ReadWrite	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi định hướng dữ liệu cổng G – DDRG

Bit	7	6	5	4	3	2	1	0	
	-	-	-	DDG4	DDG3	DDG2	DDG1	DDG0	DDRG
ReadWrite	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Địa chỉ các chân đầu vào cổng G – PING

Bit	7	6	5	4	3	2	1	0	
	-	-	-	PING4	PING3	PING2	PING1	PING0	PING
ReadWrite	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	N/A	N/A	N/A	N/A	N/A	

Chú ý rằng PORG , DDRG , và PING đều không khả dụng trong chế độ tương thích với Atmega 103 . Trong chế độ tương thích với Atmega 103 cổng G phục vụ chế độ luân phiên của nó chỉ có ( TOSC1 , TOSC2 , WR , RD , ALE )

## X . Các ngắt ngoài

Các ngắt ngoài được khởi động bằng các chân INT7:0 . Chú ý rằng , nếu được kích hoạt , các ngắt sẽ khởi động dù cho các chân INT7..0 được cấu hình như là các đầu ra . Đặc điểm này cung cấp 1 cách để sinh ra 1 phần mềm ngắt . Các ngắt ngoài có thể được khởi động bằng việc đỗ sườn xuống hoặc bắt sườn lên hoặc là một mức logic thấp . Điều này được cài đặt như đã thể hiện trong bảng đặc tính kỹ thuật của thanh ghi điều khiển các ngắt ngoài – EICRA (INT3:0) và EICRB( INT7:4). Khi các ngắt ngoài được kích hoạt được cấu hình như là 1 cấp khởi động , các ngắt sẽ khởi động chỉ cần chân được giữ ở mức thấp . Chú ý rằng sự nhận biết việc bắt sườn xuống và bắt sườn lên của các ngắt trên chân INT7:4 cần sự có mặt của 1 bộ định thời I/O , được miêu tả trong phần “ Clock System and their Distribution ” ở trang 36 . Các ngắt cấp thấp và các sườn ngắt trên INT3:0 được dò một cách dị bộ . Điều này ngụ ý rằng các ngắt này có thể được sử dụng cho việc bước (walking ) của phần cứng từ các chế độ ngủ khác hơn chế độ Idle . Bộ định thời I/O được dùng trong tất cả các chế độ ngủ trừ chế độ Idle .

Chú ý rằng nếu 1 mức ngắt đã khởi động được sử dụng cho việc đánh thức từ chế độ ngắt nguồn , mức đã thay đổi phải được giữ trong 1 khoảng thời gian để đánh thức MCU . Điều này làm cho MCU giảm sự nhạy cảm với các nhiễu . Mức đã thay đổi được lấy mẫu 2 lần bằng bộ định thời tạo dao động watchdog . Chu kỳ của bộ tạo dao động watchdog là 1  $\mu$ s (thông thường ) ở 5V và 25 độ C . Tần số của bộ tạo dao động watchdog thì phụ thuộc vào điện áp như là được chỉ ra trong phần “electrical characteristics ” ở trang 318 . MCU sẽ được đánh thức nếu đầu vào có mức cần thiết trong suốt quá trình lấy mẫu hoặc nếu nó được giữ cho đến khi kết thúc thời gian khởi động . Thời gian khởi động được xác định bằng cầu chì SUT như là được miêu tả trong phần “Clock System and their Distribution ” ở trang 36. Nếu như mức được lấy mẫu 2 lần bằng bộ tạo dao động watchdog nhưng biến mất trước khi kết thúc thời gian khởi động, MCU sẽ vẫn được đánh thức , nhưng không có ngắt nào sẽ được sinh ra . Mức cần thiết phải được giữ đủ dài cho MCU hoàn thành việc đánh thức để khởi động mức ngắt

### Thanh ghi điều khiển ngắt ngoài A – EICRA

Bit	7	6	5	4	3	2	1	0	
	ISC31	ISC30	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00	EICRA
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi này có thể không được hướng tới trong chế độ tương thích với Atmega 103 , nhưng giá trị khởi đầu của nó xác định INT3:0 như là các ngắt mức thấp , như trong Atmega 103 .

Bit 7..0 – ISC31 , ISC30 – ISC00 , ISC00 : ngắt ngoài 3 – các bit điều khiển độ nhạy 0

Các ngắt ngoài 3-0 thì được kích hoạt bằng các chân bên ngoài INT3:0 nếu như cờ I SREG và ngắt tương ứng che trong EIMSK được cài đặt . Mức và các sườn trên các chân bên ngoài cái mà kích hoạt các ngắt được xác định trong bảng 48 . Các sườn

trên INT3..INT0 thì được đăng ký 1 cách dị bộ . Các xung trên các chân INT3:0 rộng hơn độ rộng xung cực tiểu được đưa ra trong bảng 49 sẽ sinh ra một ngắt . Các xung ngắn hơn thì không đảm bảo để sinh ra 1 ngắt . Nếu mức ngắt thấp được lựa chọn , mức thấp phải được giữ cho đến khi hoàn tất lệnh đang thực thi hiện thời để sinh ra một ngắt . Nếu được kích hoạt , 1 mức ngắt đã khởi động sẽ phát sinh ra 1 yêu cầu ngắt chỉ cần chân này được giữ ở mức thấp . Khi sự thay đổi bit ISCn , 1 ngắt có thể xuất hiện . Vì vậy , nó được khuyến cáo để vô hiệu hóa đầu tiên INTn bằng việc xóa bit kích hoạt ngắt của nó trong thanh ghi EIMSK . Sau đó , bit ISCn có thể được thay đổi Cuối cùng , cờ ngắt INTn nên được xóa bằng cách viết mức logic 1 lên bit cờ ngắt của nó (INTFn ) trong thanh ghi EIFR trước khi ngắt được kích hoạt lại.

**Table 48. Interrupt Sense Control<sup>(1)</sup>**

ISCn1	ISCn0	Description
0	0	The low level of INTn generates an interrupt request.
0	1	Reserved
1	0	The falling edge of INTn generates asynchronously an interrupt request.
1	1	The rising edge of INTn generates asynchronously an interrupt request.

Note: 1. n = 3, 2, 1 or 0.

When changing the ISCn1/ISCn0 bits, the interrupt must be disabled by clearing its Interrupt Enable bit in the EIMSK Register. Otherwise an interrupt can occur when the bits are changed.

**Table 49. Asynchronous External Interrupt Characteristics**

Symbol	Parameter	Condition	Min	Typ	Max	Units
t <sub>INT</sub>	Minimum pulse width for asynchronous external interrupt			50		ns

## Thanh ghi điều khiển ngắt ngoài B – EICRB

Bit	7	6	5	4	3	2	1	0	EICRB
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7..0 – ISC71, ISC70 – ISC41, ISC40 : các bit điều khiển độ nhạy ngắt ngoài 7-4

Các ngắt ngoài 7 – 4 được kích hoạt bằng các chân ngoài INT7:4 nếu như cờ I SREG và các ngắt tương ứng che trong quá trình EIMSK được cài đặt . Mức và các sườn trên các chân ngoài cái mà kích hoạt các ngắt thì được xác định trong bảng 50 . Giá trị trên các chân INT7:4 được lấy mẫu trước khi dò các sườn xung . Nếu sườn hoặc khuỷu ngắt được lựa chọn , các xung cuối mà dài hơn 1 chu kỳ xung nhịp sẽ sinh ra 1 ngắt . Các xung ngắn hơn thì không đảm bảo để phát ra 1 ngắt . Vì vậy , tần số xung nhịp CPU có thể thấp hơn tần số XTAL nếu như bộ chia XTAL được kích hoạt . Nếu các mức ngắt thấp hơn được lựa chọn , mức thấp phải được giữ cho đến khi hoàn tất quá trình thực hiện lệnh hiện hành để sinh ra 1 ngắt . Nếu được kích hoạt , 1 mức ngắt đã khởi động sẽ sinh ra 1 yêu cầu ngắt chỉ cần nó được giữ ở mức thấp .

**Table 50. Interrupt Sense Control<sup>(1)</sup>**

ISCn1	ISCn0	Description
0	0	The low level of INTn generates an interrupt request.
0	1	Any logical change on INTn generates an interrupt request
1	0	The falling edge between two samples of INTn generates an interrupt request.
1	1	The rising edge between two samples of INTn generates an interrupt request.

Note: 1. n = 7, 6, 5 or 4.

When changing the ISCn1/ISCn0 bits, the interrupt must be disabled by clearing its Interrupt Enable bit in the EIMSK Register. Otherwise an interrupt can occur when the bits are changed.

## Thanh ghi che ngắt ngoài – EIMSK

Bit	7	6	5	4	3	2	1	0	EIMSK
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7...0 – INT7 – INT0 : kích hoạt truy vấn ngắt ngoài 7 – 0

Khi 1 bit INT7 – INT0 được ghi là 1 và bit I trong thanh ghi trạng thái (SREG) được đặt là 1 , chân ngắt ngoài tương ứng được kích hoạt . Các bit điều khiển độ nhạy ngắt trong thanh ghi điều khiển ngắt ngoài EICRA và EICRB được xác định bằng 1 trong 2 cách , ngắt ngoài được kích hoạt trên sườn lên hoặc sườn xuống hoặc mức độ nhạy ( level sensed ). Hoạt động trên bất cứ một chân nào sẽ khởi động một yêu cầu ngắt dù cho chân này được kích hoạt như là một đầu ra . Điều này cung cấp 1 cách của việc sinh ra phần mềm ngắt .

## Thanh ghi cờ ngắt ngoài EIFR (external interrupt flag register )

Bit	7	6	5	4	3	2	1	0	EIFR
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7...0 – INTF7 – INTF0 : cờ ngắt ngoài 7 – 0

Khi một sườn hoặc thay đổi biến logic trên chân INT7:0 khởi động một yêu cầu ngắt .. INTF7:0 sẽ được cài đặt là 1 .Nếu bit I trong SREG và bit kích hoạt ngắt tương ứng , INT7:0 trong thanh ghi EIMSK được cài đặt là 1 , MCU sẽ nhảy tới vec to ngắt. Cờ bị xóa khi một chương trình con phục vụ ngắt được thực thi . Như một sự lựa chọn, cờ có thể bị xóa bằng việc viết mức logic 1 lên nó . Các cờ thì luôn được xóa khi mà INT7:0 được cấu hình như là một mức ngắt . Chú ý rằng khi truy nhập vào chế độ sleep với các ngắt INT3:0 đã vô hiệu hóa , các bộ đếm đầu vào trên các chân này sẽ bị vô hiệu hóa . Điều này có thể gây ra 1 sự thay đổi biến logic trong tín hiệu bên trong cái mà sẽ được cài đặt các cờ INT3:0 . Xem thêm "Digital Input Enable and Sleep Modes ở trang 70 để biết thêm thông tin

## XI . 8-bit Timer/Counter 0 với PWM and Asynchronous Operation ( Timer/Counter 8 bit với PWM và điều khiển dị bộ )

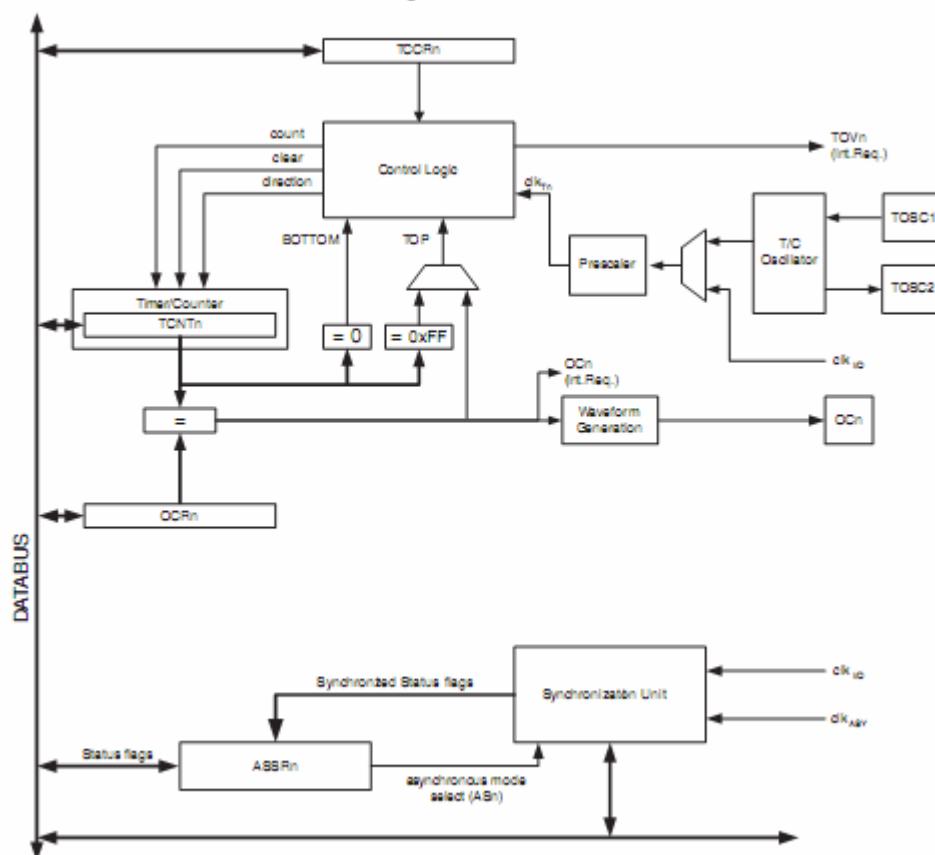
Timer/Counter0 là một module Timer/Counter 8 bit , kênh đơn , đa năng dùng chung . Các đặc điểm chính là :

- Bộ đếm kênh đơn
- Xóa Timer trên bộ so sánh tương ứng ( tự động tải lại )
- Không có nhiễu sọc ngang , điều chế độ rộng xung đúng pha (PWM)
- Máy phát tần số
- Bộ đếm gộp trước xung nhịp 10 bit
- Dòng tràn (overflow) và các nguồn ngắn ghép so sánh (TOV0 và OCF0 )
- Cho phép bộ định thời từ bên ngoài 32kHz đồng hồ thạch anh phụ thuộc vào xung nhịp I/O

### Tổng quan

Một sơ đồ khối đã rút gọn của 1 Timer/Counter 8bit được đưa ra trong hình 34 . Về sự sắp đặt hiện tại của các chân I/O , tham khảo phân cầu hình chân ở trang 2 . CPU có thể truy nhập các thanh ghi I/O , bao gồm các bit I/O và các chân I/O , được in đậm . Thanh ghi I/O của thiết bị xác định và vị trí các bit được liệt kê trong phần “8-bit Timer/Counter Register Description “ ở trang 104

Figure 34. 8-bit Timer/Counter Block Diagram



## Các thanh ghi

Timer/Counter (TCNT0) và các thanh ghi so sánh đầu ra (OCR0) là các thanh ghi 8 bit. Các tín hiệu Yêu cầu ngắt (viết tắt là Int.Req.) thì đều được nhìn thấy trong thanh ghi cờ ngắt Timer (TIFR). Tất cả các ngắt đều được che riêng với thanh ghi che ngắt Timer (TIMSK). TIFR và TIMSK đều không được chỉ ra trong hình từ đó các thanh ghi này được chia sẻ bằng các bộ phận timer khác.

Timer/Counter có thể bị khóa bên trong, như bộ đếm gộp trước hoặc bị khóa một cách không đồng bộ khỏi các chân TOSC1/2, như được miêu tả chi tiết trong các phần sau. Quá trình điều khiển dị bộ được điều khiển bởi thanh ghi trạng thái dị bộ (ASSR). Khối logic lựa chọn xung nhịp điều khiển cái mà nguồn xung nhịp Timer/Counter sử dụng để làm tăng hoặc giảm giá trị của nó. Timer/Counter thì không hoạt động khi không có nguồn phát xung nhịp nào được lựa chọn. Đầu ra từ bộ định thời lựa chọn mức logic được hướng dẫn như là một xung nhịp timer ( $clk_{T0}$ )

Thanh ghi so sánh đầu ra (OCR0) lưu trong bộ đếm kép thì được so sánh với giá trị của Timer/Counter tại tất cả các thời gian. Kết quả của phép so sánh có thể được sử dụng máy phát dạng sóng để phát ra xung PWM hoặc đầu ra tần số biến thiên trên chân so sánh đầu ra OC0). Xem thêm “Output Compare Unit” trên trang 95 để biết thêm chi tiết. Bộ so sánh ghép sự kiện cũng sẽ cài đặt cờ so sánh (OCF0) cái mà có thể được sử dụng để sinh ra 1 yêu cầu ngắt so sánh đầu ra

## Các định nghĩa

Nhiều thanh ghi và các bit tham khảo trong tài liệu này được viết theo mẫu chung. Một trường hợp thấp “n” được thay thế cho số thứ tự của Timer/Counter, trong trường hợp này là 0. Tuy nhiên, khi sử dụng thanh ghi hoặc bit xác định trong 1 chương trình, một mẫu chính xác phải được sử dụng (ví dụ TCNT0 cho việc truy nhập Timer/Counter 0 đếm giá trị)

Định nghĩa ở bảng 51 thì cũng được sử dụng một cách rộng rãi xuyên suốt tài liệu này

<b>BOTTOM</b>	The counter reaches the BOTTOM when it becomes zero (0x00).
<b>MAX</b>	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
<b>TOP</b>	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR0 Register. The assignment is dependent on the mode of operation.

## Các nguồn xung nhịp Timer/Counter

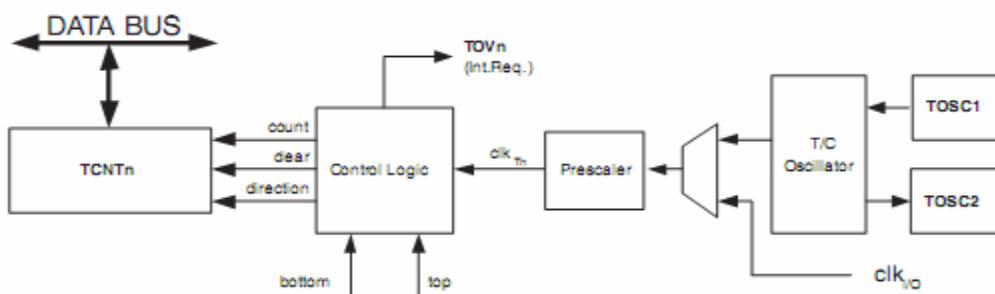
Timer/Counter có thể bị khóa bằng việc 1 bộ đồng bộ bên trong hoặc một nguồn xung nhịp dị bộ bên ngoài. Nguồn xung nhịp  $clk_{T0}$  thì bằng sự tính toán mặc định lên xung nhịp MCU,  $clk_{I/O}$ . Khi mà bit AS0 trong thanh ghi ASSR được viết mức logic 1, nguồn phát xung nhịp được tạo ra từ bộ tạo dao động của Timer/Counter được kết nối

với TOSC1 và TOSC2. Để thêm chi tiết về quá trình điều khiển bộ xem thêm “Asynchronous Status Register – ASSR ở trang 107, để biết thêm chi tiết về nguồn phát xung nhịp và bộ đếm gộp trước xem phần “Timer/Counter Prescaler” ở trang 110

## Đơn vị của bộ đếm

Phần chính của Timer/Counter 8 bit thì có thể được lập trình thành phần bộ đếm 2 hướng . Hình 35 chỉ ra 1 sơ đồ khối của 1 counter và môi trường xung quanh của nó

**Figure 35.** Counter Unit Block Diagram



### Mô tả tín hiệu ( tín hiệu bên trong )

Count	tăng hoặc giảm TCNT0 bằng 1
Direction	lựa chọn giữa việc tăng và giảm
Clear	xóa TCNT0 (đặt tất cả các bit là 0 )
Clk <sub>T0</sub>	xung nhịp Timer/Counter
Top	được chú ý rằng TCNT0 vươn tới giá trị cực đại
Bottom	được chú ý rằng TCNT0 vươn tới giá trị cực tiểu

Phụ thuộc vào chế độ điều khiển được sử dụng , bộ đếm bị xóa , được làm tăng hoặc giảm tại mỗi xung nhịp thời gian (clk<sub>T0</sub>) . clk<sub>T0</sub> có thể được sinh ra từ 1 nguồn phát xung nhịp bên ngoài hoặc bên trong , được lựa chọn bằng các bit lựa chọn xung nhịp (CS02:0) . Khi không có nguồn xung nhịp nào được lựa chọn (CS02:0=0) timer được dừng lại . Tuy nhiên , giá trị TCNT0 có thể được truy nhập bằng CPU , bất chấp việc xung clk<sub>T0</sub> được đưa ra hay không . 1 CPU viết ghi đè lên (có quyền ưu tiên trên ) xóa tất cả các counter hoặc là các quá trình điều khiển sự đếm .

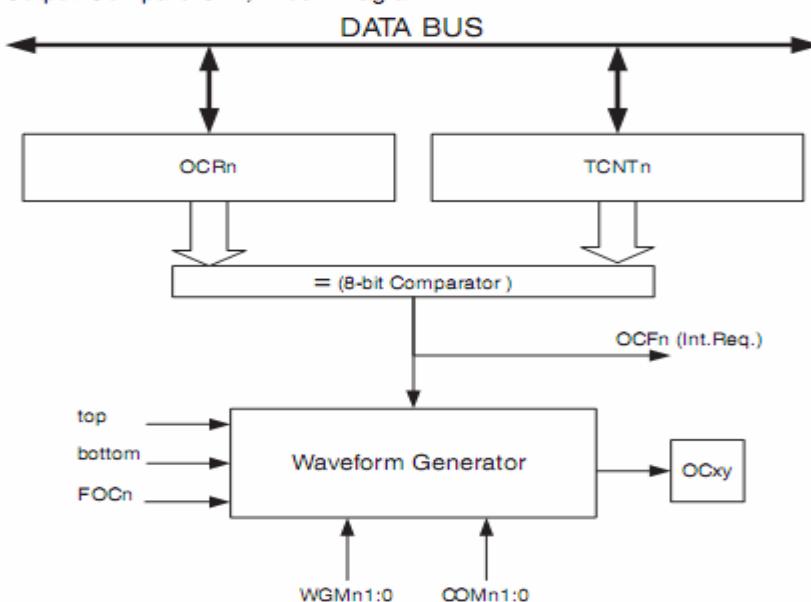
Sự đếm các chuỗi được xác định rõ bằng việc cài đặt của các bit WGM01 và WGM00 được đặt trong thanh ghi điều khiển Timer/Counter (TCCR0). Có các kết nối đóng giũa cách các bộ đếm vận hành và cách các dạng sóng được sinh ra trên đầu ra so sánh đầu ra OC0 . Để biết thêm chi tiết về sự đếm chuỗi hoàn thiện và sự sinh ra các dạng sóng , xem thêm phần “Modes of Operation “ trên trang 98

Cờ báo Dòng tràn của Timer/Counter được cài đặt theo chế độ điều khiển được lựa chọn bằng các bit WGM01:0 . TOV0 có thể được sử dụng cho việc phát ra các ngắt của CPU

### Các bộ phận so sánh đầu ra (Output Compare Unit )

Một bộ so sánh 8 bit liên tục so sánh TCNT0 với thanh ghi so sánh đầu ra (OCR0). Bất cứ nơi nào TCNT0 tính toán OCR0 , bộ so sánh báo hiệu 1 match . Một match sẽ cài đặt cờ báo đầu ra so sánh (OCF0) ở chu kỳ xung nhịp timer tiếp theo . Nếu được kích hoạt (OCIE0=1) , cờ báo đầu ra so sánh sinh ra 1 ngắt đầu ra so sánh . Cờ OCF0 bị xóa 1 cách tự động khi mà ngắt đã được thực thi . Như một sự lựa chọn , cờ OCF0 có thể được xóa bằng phần mềm bằng cách viết mức logic 1 lên vị trí bit I/O của nó . Bộ phát dạng sóng sử dụng tín hiệu ghép để sinh ra 1 đầu vào theo chế độ điều khiển được cài đặt bởi các bit WGM01:0 và các bit chế độ đầu ào so sánh (COM01:0) . Tín hiệu max và bottom thì được sử dụng bởi máy phát dạng sóng cho quá trình xử lí trong trường hợp đặc biệt của giá trị cực biên trong một vài chế độ điều khiển “modes of Operation ” ở trang 98 . Hình 36 chỉ ra sơ đồ khối của bộ phận so sánh đầu ra .

Figure 36. Output Compare Unit, Block Diagram



Thanh ghi OCR0 thì được lưu vào bộ nhớ đệm kép khi sử dụng bất cứ xung nào của chế độ điều chế độ rộng xung (PWM) ..Về các chế độ điều khiển bình thường và chế độ so sánh trên timer xóa , bộ đệm kép bị vô hiệu hóa . Việc cập nhật bộ đệm kép đồng bộ hóa của thanh ghi so sánh OCR0 hoặc là trên top hoặc là bottom của chuỗi đếm . Việc đồng bộ hóa dữ liệu ngăn cản sự xuất hiện của các độ dài lẻ ( Odd-length ) và các xung PWM không đối xứng , bằng cách tạo ra các đầu ra không có nhiễu sọc ngang .

Dữ liệu Thanh ghi OCR0 có thể dường như phức tạp , nhưng không phải trong trường hợp này . Khi mà bộ đệm kép được kích hoạt , CPU truy nhập vào bộ đệm của thanh ghi OCR0 , và nếu bộ đệm kép bị vô hiệu hóa thì CPU sẽ truy nhập trực tiếp vào OCR0

## Đầu ra so sánh cường bức

trong chế độ khôn phải chế độ phát dạng sóng PWM , cổng ra ghép của bộ so sánh có thể bị cưỡng ép bằng việc viết là 1 lên bit so sánh đầu ra cường bức (Force Output Compare bit – FOC0). Việc cưỡng ép so sánh ghép sẽ không cài đặt cờ OCF0 hoặc tải lại/xóa Timer , nhưng chân OC0 sẽ được cập nhật lại như là 1 ghép so sánh thực đã xuất hiện (các bit COM01:0 cài đặt xác định chân OC0 được cài đặt , bị xóa hoặc dịch chuyển )

### Sự khóa ghép so sánh bằng việc viết bit TCNT0 – Compare Match Blocking by TCNT0 Write

Tất cả các quá trình viết của thanh ghi tới thanh ghi TCNT0 sẽ khóa bất cứ ghép so sánh nào cái mà xuất hiện trong chu kì xung nhịp kế tiếp , thậm chí cả khi Timer đã bị dừng lại . Đặc điểm này cho phép OCR0 được khởi tạo đến các giá trị giống như TCTN0 mà không khởi động 1 ngắt khi mà đồng hồ Timer/Counter được kích hoạt

### Using the Output Compare Unit : Việc sử dụng bộ phận so sánh đầu ra

Từ khi việc viết TCTN0 trong bất cứ chế độ điều khiển nào sẽ khóa tất cả các ghép so sánh cho 1 chu kì xung nhịp timer , có những nguy cơ được hàm chúa khi sự thay đổi TCNT0 khi sử dụng kênh so sánh đầu ra , điều này phụ thuộc vào Timer/Counter đang chạy hay không . Nếu giá trị được viết lên TCNT0 bằng giá trị OCR0, ghép so sánh sẽ bị lỗi , kết quả là sự phát dạng sóng không đúng. Một cách tương tự , không được viết giá trị của TCNT0 bằng với giá trị BOTTOM khi mà counter đang đếm xuống ( đếm lùi )

Sự cài đặt OC0 nên được thi hành trước khi việc cài đặt thanh ghi định hướng dữ liệu cho các chân cổng lên cổng ra . Cách dễ nhất của việc cài đặt giá trị của OC0 là sử dụng đầu ra so sánh cường bức (FOC0) phân tích bit trong chế độ thường . Thanh ghi OC0 giữ giá trị của nó dù cho khi thay đổi giữa các chế độ phát dạng sóng (wareform generation modes )

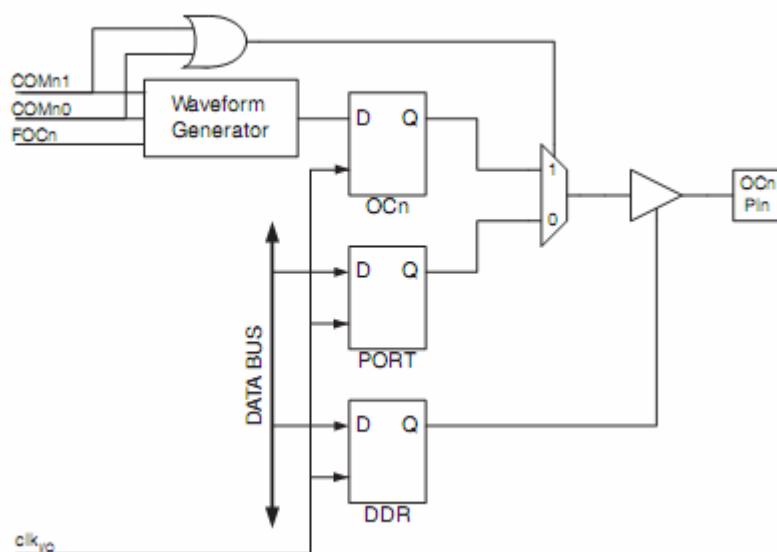
Phải nhận thức rằng các bit COM01:0 thì không được ghi vào bộ đệm kép cùng nhau với các giá trị so sánh . Việc thay đổi các bit COM01:0 sẽ tạo hiệu quả ngay lập tức

### Thành phần đầu ra ghép so sánh – Compare Match Output Unit

Các bit chế độ đầu ra so sánh (COM01:0 ) có hai chức năng . Máy phát dạng sóng sử dụng các bit COM01:0 để xác định trạng thái đầu ra so sánh (OC0 ) tại ghép so sánh tiếp theo (compare match ) . các bit COM01:0 cũng điều khiển các chân nguồn đầu ra OC0. Hình 37 chỉ là sơ đồ đã rút gọn của các bit bị hư hỏng của việc cài đặt bit COM01:0 . Thanh ghi I/O , các bit I/O , và các chân I/O trong hình vẽ được in đậm .

Chỉ có những phần của thanh ghi điều khiển cổng I/O chung (DDR và PORT) cái mà bị hư hỏng bởi các bit COM01:0 được chỉ ra. Khi tham khảo trạng thái OC0, sự tham khảo cho thanh ghi OC0 bên trong không phải là chân OC0.

**Figure 37. Compare Match Output Unit, Schematic**



Chức năng của các chân I/O chung được ghi đè bởi bit so sánh đầu ra (OC0) từ máy phát dạng sóng nếu một trong 2 bit COM01:0 được cài đặt. Tuy nhiên, hướng của chân OC0 (đầu vào hoặc đầu ra thì vẫn được điều khiển bởi thanh ghi định hướng dữ liệu DDR cho các chân cổng. Bit thanh ghi định hướng dữ liệu cho chân OC0 (DDR\_OC0) phải được cài đặt như là đầu ra trước khi giá trị OC0 được nhìn thấy trên chân. Chức năng ghi đè cổng thì phụ thuộc vào chế độ phát dạng sóng

Thiết kế của chân so sánh đầu ra logic cho phép việc khởi tạo của trạng thái OC0 trước khi đầu ra được kích hoạt. Chú ý rằng vài việc cài đặt các bit COM01:0 là dự trữ cho các chế độ biến cố đã biết của quá trình điều khiển. Xem “8bit Timer/Counter Register Description ở trang 104

### Chế độ đầu ra so sánh và máy phát dạng sóng

Máy phát dạng sóng sử dụng các bit COM01:0 khác nhau trong các chế độ bình thường, CTC, PWM. Với tất cả các chế độ, việc cài đặt COM01:0 nói cho máy phát dạng sóng rằng không có hành động nào trên thanh ghi OC0 được tiến hành trên ghép so sánh kế tiếp. Về phần các ảnh hưởng đầu ra so sánh trong các chế độ không phải PWM tham khảo bảng 53 ở trang 105. Cho các chế độ PWM nhanh, tham khảo bảng 54 trang 105, và PWM đúng pha tham khảo bảng 55 trang 106

Một thay đổi của trạng thái các bit COM01:0 sẽ có hiệu lực tại ghép so sánh đầu tiên sau khi các bit được viết. Cho các chế độ không phải PWM, hoạt động có thể được cường ép để có hiệu quả ngay lập tức bằng việc sử dụng các bit phân tích FOC0

## Các chế độ điều khiển

Các chế độ của quá trình điều khiển , ví dụ như trạng thái của Timer/Counter và các chân so sánh đầu ra thì được xác định bằng cách kết hợp chế độ phát dạng sóng (WGM01:0 ) và chế độ đầu ra so sánh các bit (COM01:0). Các bit chế độ đầu ra so sánh thì không có ảnh hưởng đến việc đếm các chuỗi trong khi các bit chế độ phát dạng sóng làm việc . Các bit COM01:0 điều khiển đầu ra PWM được sinh ra nên được khuyêch đại hoặc không (inverted or non-inverted PWM) . Về các chế độ không phải PWM , các bit COM01:0 điều khiển hoặc đầu ra nên được cài đặt , bị xóa , hoặc bị dịch chuyển tại 1 ghép so sánh ( xem bảng Compare Match Output Unit “ở trang 97 ” )

Để biết thêm chi tiết tham khảo “Timer/Counter Timing Diagrams “ ở trang 102

## Chế độ bình thường

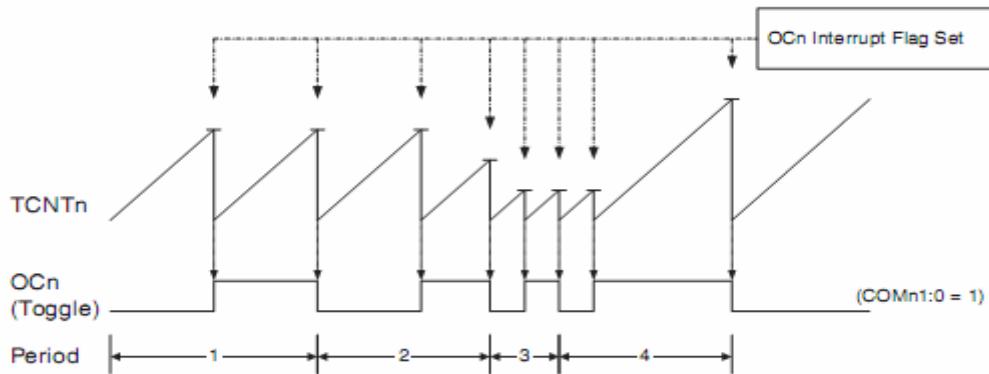
Chế độ đơn giản nhất của quá trình điều khiển là chế độ bình thường (normal ) ( $WGM01:0 = 0$  ) . Trong chế độ này việc đếm định hướng thì luôn hướng lên trên ( Incrementing ) , và không có xóa bộ đếm nào được thi hành . Bộ đếm đơn giản tràn qua khi nó vượt quá giá trị cực đại 8 bit của nó ( $TOP=0xFF$ ) và sau đó khởi động từ mức bottom ( $0x00$ ) . Trong quá trình điều khiển thông thường cờ báo tràn Timer/Counter (TOV0) sẽ được cài đặt trong các chu kì xung nhịp giống nhau như là TCNT0 trở thành 0. Cờ TOV0 trong trường hợp này thực hiện giống như 1 bit thứ 9 , ngoại lệ là nó chỉ được cài đặt , không bị xóa . Tuy nhiên , được nối với các ngắt tràn Timer cái mà tự động xóa cờ TOV0 , độ chính xác của timer có thể được tăng lên bằng phần mềm . Không có trường hợp đặc biệt nào để xét đến trong chế độ bình thường , 1 giá trị bộ đếm mới có thể được viết vào bất cứ thời gian nào .

Bộ phận đầu vào so sánh có thể được sử dụng để phát ra các ngắt tại một vài thời gian được đưa ra trước . Việc sử dụng so sánh đầu ra để phát dạng sóng trong chế độ bình thường thì không được khuyến cáo , từ khi điều này sẽ chiếm quá nhiều thời gian của CPU

## Xóa timer trên chế độ ghép so sánh (Clear Timer on Compare Match Mode CTC )

Trong chế độ xóa Timer trong chế độ so sánh hoặc là CTC mode ( $WGM01:0=2$ ), thanh ghi OCR0 được sử dụng để điều khiển độ chính xác của bộ đếm . Trong chế độ CTC bộ đếm được xóa là 0 khi giá trị của bộ đếm (TCNT0) tương ứng với OCR0 . OCR0 xác định giá trị định của bộ đếm , do đó cũng là thay đổi độ chính xác của nó . Chế độ này cho phép điều khiển lớn hơn của tần số đầu ra so sánh . Nó cũng rút gọn quá trình điều khiển của việc đếm các sự kiện bên ngoài

Giản đồ thời gian cho chế độ CTC được chỉ ra trong hình 38 . Giá trị của bộ đếm(TCNT0) được tăng cho đến khi 1 ghép so sánh xuất hiện giữa TCNT0 và OCR0 và sau đó bộ đếm TCNT0 bị xóa .

**Figure 38. CTC Mode, Timing Diagram**

Một ngắt có thể được sinh ra mỗi lần mà giá trị của bộ đếm vươn tới giá trị TOP bằng việc sử dụng cờ OCF0 . Nếu như ngắt được kích hoạt , các chương trình con điều khiển ngắt có thể được sử dụng để cập nhật giá trị TOP . Tuy nhiên , việc thay đổi TOP đến 1 giá trị đóng Bottom khi mà bộ đếm đang chạy với 0 hoặc 1 giá trị thấp của bộ đếm gộp trước phải được làm với sự cẩn thận từ khi chế độ CTC có đặc điểm là bộ đếm kép . Nếu giá trị mới được viết lên OCR0 thấp hơn giá trị hiện thời của TCNT0 , bộ đếm sẽ bị lỗi ghép so sánh . Bộ đếm sau đó phải đếm từ giá trị cực đại của nó (0xFF) và bọc xung quanh điểm khởi đầu 0x00 trước khi ghép so sánh có thể xuất hiện .

Về việc phát ra một dạng sóng đầu ra trong CTC mode , đầu ra OC0 có thể được cài đặt để di chuyển mức logic của nó trên mỗi ghép so sánh bằng việc cài đặt các bit chế độ đầu ra so sánh tới chế độ Toggle (COM01:0=1 ) . Giá trị OC0 sẽ không được nhìn thấy ở trên chân cổng trừ phi việc định hướng dữ liệu cho các chân được cài đặt tới đầu ra . Dạng sóng được phát ra sẽ có một tần số cực đại của  $f_{OC0} = f_{clk\_I/O}/2$  khi mà OCR0 được cài đặt là 0 (0x00) . Tần số dạng sóng được xác định bằng công thức dưới đây :

$$f_{OCn} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRn)}$$

Biến N được trình bày theo tỉ lệ cho trước (1,8, 32,64,128,256, hoặc 1024) Cho chế độ thường của quá trình điều khiển , cờ TOV0 được cài đặt trong các chu kỳ xung nhịp khác nhau cái mà bộ đếm đếm từ MAX đến 0x00

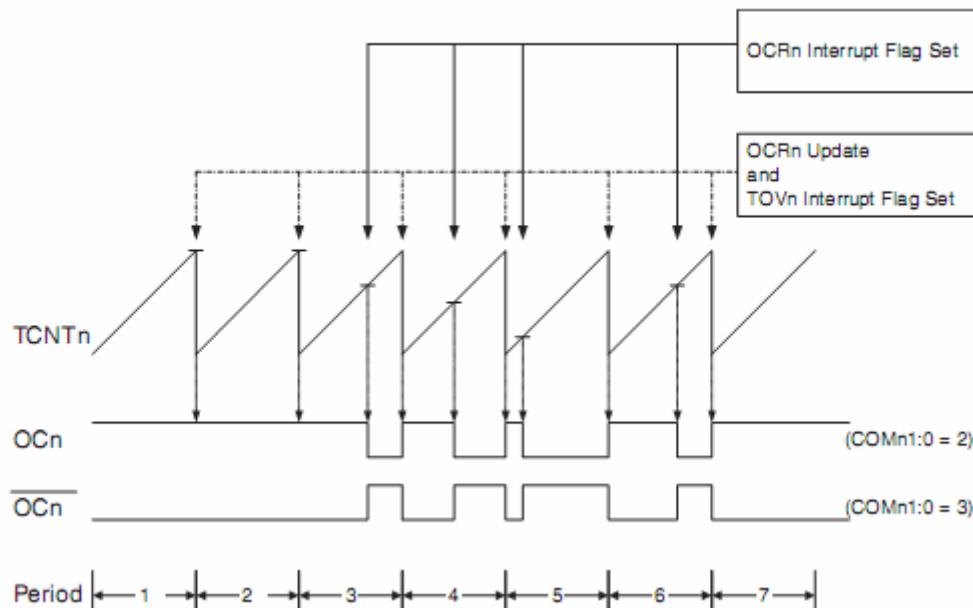
### Chế độ Fast PWM : xung PWM cố định

Độ rộng xung điều chế cố định hoặc chế độ Fast PWM (WGM01:0 = 3 ) cung cấp 1 sự lựa chọn phát dạng sóng PWM tần số cao . Chế độ fast PWM khác các lựa chọn PWM khác bằng các chế độ điều khiển sườn đơn . Bộ đếm đếm từ mức BOTTOM đến MAX sau đó khởi động lại từ BOTTOM . Trong chế độ đầu ra so sánh không đảo , bit so sánh đầu ra (OC0) bị xóa trên ghép so sánh giữa TCNT0 và OCR0 , và đặt ở BOTTOM . Để tương ứng với chế độ điều khiển sườn đơn , quá trình điều khiển tần số của chế độ fast PWM có thể được cao gấp đôi như là các pha đúng của chế độ PWM cái mà được sử dụng quá trình điều khiển 2 sườn . Tần số cao này làm cho chế độ fast PWM phù hợp với sự điều chỉnh nguồn , sự chỉnh lưu và các ứng dụng

DAC . Tần số cao này cho phép các thành phần vật lý cỡ nhỏ (cuộn dây , tụ điện ) và vì vậy giảm giá thành của hệ thống .

Trong chế độ Fast PWM , bộ đếm được làm tăng cho đến khi giá trị của bộ đếm tương ứng với giá trị MAX . Bộ đếm thì sau đó được xóa tại chu kỳ xung nhịp timer bên dưới . Giản đồ thời gian cho chế độ Fast PWM được đưa ra trên hình 39 . Giá trị của TCNT0 trong giản đồ thời gian chỉ ra như là 1 biểu đồ minh họa cho quá trình điều khiển sườn đơn . Giản đồ bao gồm các đầu ra PWM đảo và không đảo .Đường nằm ngang nhỏ đánh dấu trên các sườn TCNT0 được đưa ra các ghép so sánh giữa OCR0 và TCNT0

**Figure 39. Fast PWM Mode, Timing Diagram**



Cờ báo tràn Timer/Counter TOV0 được cài đặt mỗi lần bộ đếm tiến tới giá trị MAX nếu như ngắt được kích hoạt , chương trình con điều khiển ngắt có thể được sử dụng cho việc cập nhật các giá trị so sánh

Trong chế độ Fast PWM , bộ phận so sánh cho phép việc phát ra các dạng sóng PWM trên chân OC0 . Việc cài đặt các bit COM01:0 lên 2 sẽ gây ra 1 xung PWM không đảo và 1 đầu ra PWM đảo có thể được phát ra bằng việc cài đặt COM01:0 lên 3 ( xem bảng 54 trang 105 ) . Giá trị thực của OC0 sẽ chỉ được nhìn thấy trên chân cổng nếu định hướng dữ liệu cho chân cổng được cài đặt như là cổng ra . Dạng sóng PWM được phát ra bằng việc cài đặt hoặc (xóa ) thanh ghi OC0 tại ghép so sánh giữa OCR0 và TCNT0 , và việc xóa (hoặc cài đặt ) thanh ghi OC0 tại thời gian chu kỳ xung nhịp bộ đếm được xóa ( thay đổi từ MAX đến BOTTOM )

Tần số PWM cho đầu ra có thể được tính bằng công thức sau đây:

$$f_{OCnPWM} = \frac{f_{clk\_I/O}}{N \cdot 256}$$

Biến N được đưa ra theo tỉ lệ xích (1 , 8 , 32 , 64 , 128, 256, 1024 )

Giá trị cực đại của thanh ghi OCR0 được đưa ra trong các trường hợp đặc biệt khi mà sự phát ra một dạng sóng đầu ra PWM trong chế độ Fast PWM . Nếu như OCR0

được cài đặt bằng giá trị BOTTOM , đầu ra sẽ là các đỉnh nhọn và hẹp cho mỗi một chu kì xung nhịp MAX+1 . Việc cài đặt OCR0 bằng MAX sẽ là hiệu quả trong 1 đầu ra có giá trị cao hoặc thấp là hằng số ( sự phụ thuộc vào độ phân cực của đầu ra được cài đặt bằng các bit COM01:0 )

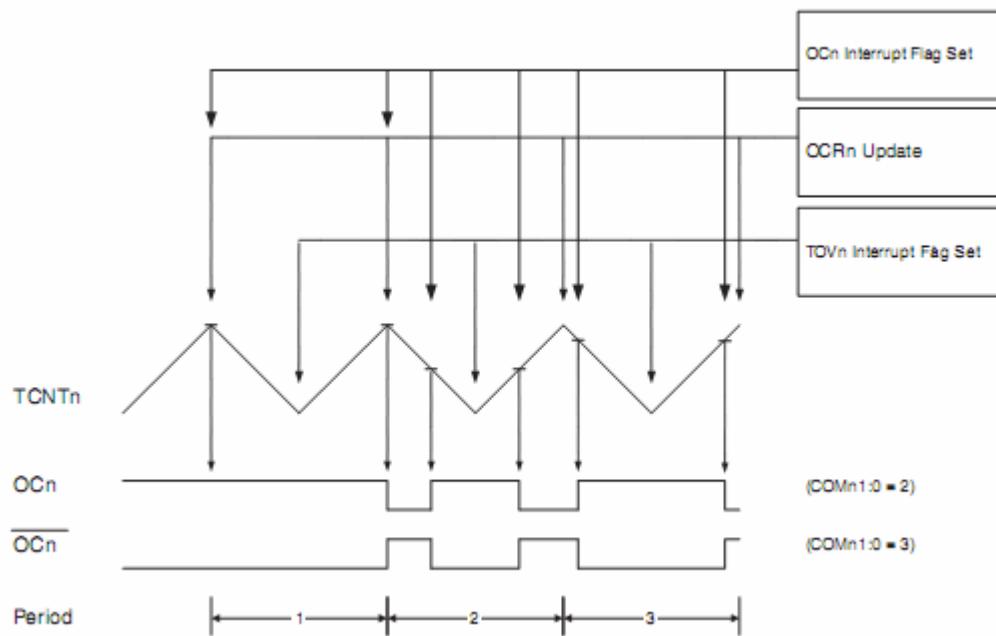
Một tần số (với 50% tải làm việc ) của dạng sóng đầu ra trong chế độ PWM có thể đạt được bằng việc cài đặt OC0 di chuyển các mức logic của nó trên mỗi ghép so sánh (COM01:0 =1) . Dạng sóng được phát ra sẽ có 1 giá trị cực đại của  $f_{OC0} = f_{clk\_I/O}/2$  khi mà OCR0 được cài đặt là 0 . Đặc điểm này thì tương tự như sự di chuyển OC0 trong chế độ CTC . , ngoại trừ đặc điểm của bộ đếm kép của bộ phận so sánh đầu ra được kích hoạt trong chế độ fast PWM

### **Chế độ PWM đúng pha**

Chế độ PWM đúng pha (WGM1:0 =1) cung cấp 1 sự lựa chọn phát các dạng sóng PWM đúng pha có độ chính xác cao . Chế độ đúng pha PWM cơ bản là quá trình điều khiển xung đơn . Bộ đếm đếm lặp lại từ BOTTOM đến MAX và sau đó từ MAX đến BOTTOM . Trong chế độ đầu ra so sánh không đảo , so sánh đầu ra (OC0) được xóa trên các ghép so sánh giữa TCNT0 và OCR0 trong quá trình đếm lên , và được cài đặt trên ghép so sánh trong khi đếm xuống . Trong chế độ so sánh đầu ra đảo , quá trình điều khiển được đảo . Quá trình điều khiển 2 sườn có tần số hoạt động lớn nhất thấp hơn quá trình điều khiển sườn đơn . Tuy nhiên , so đặc tính đối xứng của các chế độ PWM 2 sườn , các chế độ này được ưa thích hơn trong các ứng dụng điều khiển động cơ .

Độ chính xác của PWM cho chế độ PWM đúng pha là 8 bit bền vững . Trong chế độ PWM đúng pha bộ đếm được làm tăng cho đến khi giá trị của bộ đếm tương ứng với MAX khi mà bộ đếm tiến tới MAX , nó thay đổi hướng đếm . Giá trị TCNT0 sẽ bằng với MAX cho 1 chu kì xung nhịp thời gian . Giản đồ thời gian cho chế độ PWM đúng pha được đưa ra trong hình 40 . Giá trị TCNT0 trong giản đồ thời gian chỉ ra như là 1 biểu đồ minh họa cho quá trình điều khiển 2 sườn . Giản đồ bao gồm các đầu vào PWM đảo và không đảo . Đường nằm ngang nhỏ đánh dấu sườn TCNT0 được đưa ra trên ghép so sánh giữa OCR0 và TCNT0.

Figure 40. Phase Correct PWM Mode, Timing Diagram



Cờ báo tràn Timer/Counter được đặt mỗi lần bộ đếm tiến tới BOTTOM . Cờ ngắt có thể được sử dụng để sinh ra 1 ngắt mỗi lần bộ đếm tiến tới giá trị BOTTOM .

Trong chế độ PWM đúng pha , bộ phận so sánh cho phép phát sinh ra các dạng sóng PWM trên chân OC0 . Việc cài đặt các bit COM01:0 lên 2 sẽ gây ra 1 xung PWM không đảo. Một đầu ra PWM đảo có thể được sinh ra bằng việc cài đặt COM01:0 lên 3 ( xem bảng 55 trang 106 ) . Giá trị thực sự của OC0 sẽ chỉ được nhìn thấy trên chân công nếu như sự định hướng dữ liệu cho các chân công được cài đặt như là đầu ra . Dạng sóng PWM được sinh ra bằng cách xóa (hoặc cài đặt ) thanh ghi OC0 tại ghép so sánh giữa OCR0 và TCNT0 khi mà bộ đếm tăng và việc cài đặt ( hoặc xóa ) thanh ghi OC0 tại ghép so sánh giữa OCR0 và TCNT0 khi mà bộ đếm giảm . Tần số của PWM cho đầu ra khi việc sử dụng chế độ PWM đúng pha có thể được tính toán bằng công thức dưới đây :

$$f_{OCnPCPWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

Biến N được đưa ra theo tỉ lệ xích (1 , 8 , 32, 64 , 128 , 256, 1024 )

Giá trị cực biên của thanh ghi OCR0 được đưa ra trong trường hợp đặc biệt khi mà sự sinh ra 1 đầu ra dạng sóng PWM trong chế độ PWM đúng pha . Nếu như OCR0 được cài đặt bằng BOTTOM , đầu ra sẽ tiếp tục ở mức thấp và nếu nó cài đặt bằng MAX thì đầu ra sẽ tiếp tục ở mức cao cho chế độ PWM không đảo . Về chế độ PWM đảo thì đầu ra sẽ có các giá trị logic đối lập

Ở mỗi điểm bắt đầu của chu kỳ 2 trong hình 40 Ocn có sự chuyển đổi từ cao xuống thấp mặc dù không có ghép so sánh nào . Điểm đánh dấu của sự thay đổi này được đảm bảo đối xứng xung quanh BOTTOM . Có hai trường hợp được đưa ra 1 sự chuyển đổi mà không ghép so sánh :

- OCR0 thay đổi giá trị của nó từ MAX , giống như hình 40 . Khi mà giá trị của OCR0 là MAX , giá trị của chân Ocn thì giống như kết quả của việc đếm xuống

ghép so sánh . Để đảm bảo tính đối xứng xung quanh BOTTOM , giá trị OCn tại MAX phải được tương ứng với kết quả của việc đếm lên ghép so sánh .

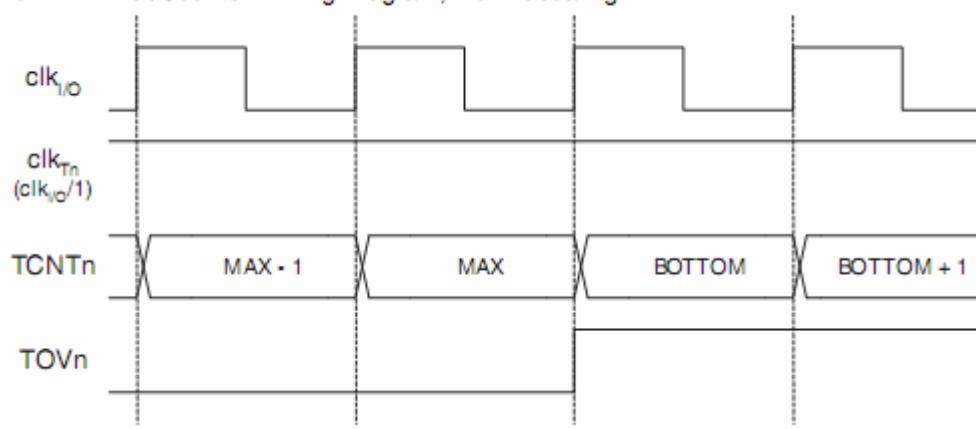
- Timer bắt đầu đếm từ giá trị cao hơn về giá trị thấp hơn trong OCR0 , và vì lí do này các lỗi của ghép so sánh và do đó OCn thay đổi cái mà sẽ xảy ra trong chiều lên .

## Giản đồ thời gian của Timer/Counter

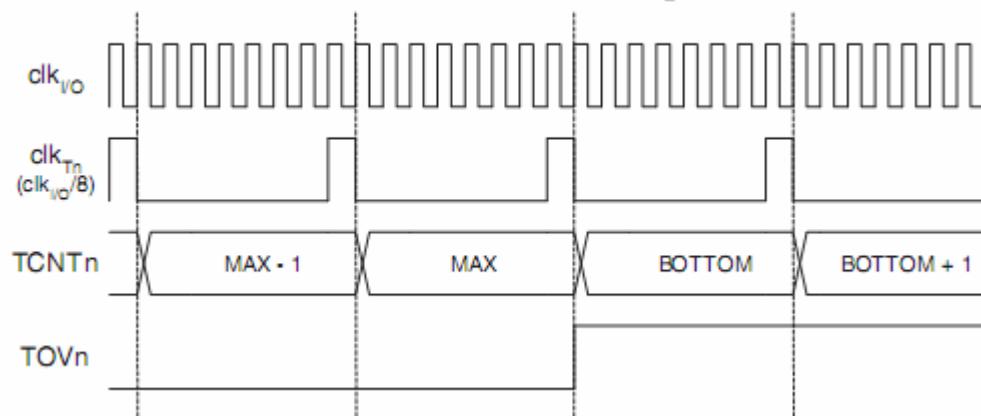
Hình 41 và hình 42 bao gồm dữ liệu thời gian cho quá trình điều khiển Timer/Counter . Timer/Counter thì được thiết kế đồng bộ và các xung nhịp timer ( $clk_{T_0}$ ) do vậy được chỉ ra như là 1 xung nhịp kích hoạt tín hiệu. Hình này chỉ ra chuỗi đếm lên các giá trị MAX . Hình 43 và 44 chỉ ra các dữ liệu thời gian giống nhau , nhưng với các bộ đếm gộp trước được kích hoạt . Các hình cũng minh họa khi các cờ ngắt được cài đặt .

Các hình bên dưới chỉ ra Timer/Counter trong chế độ đồng bộ , và xung nhịp timer ( $clk_{T_0}$ ) thì do vậy được chỉ ra như là 1 xung kích hoạt các tín hiệu . Trong chế độ dị bộ ,  $clk_{I/O}$  nên được thay thế bằng xung nhịp của bộ tạo dao động Timer/Counter . Các hình bao gồm thông tin về khi các cờ báo ngắt được cài đặt . Hình 44 bao gồm dữ liệu thời gian cho quá trình điều khiển Timer/Counter cơ bản . Hình này cũng chỉ ra các chuỗi đếm đóng lên giá trị MAX trong tất cả các chế độ khác hơn chế độ PWM đúng pha

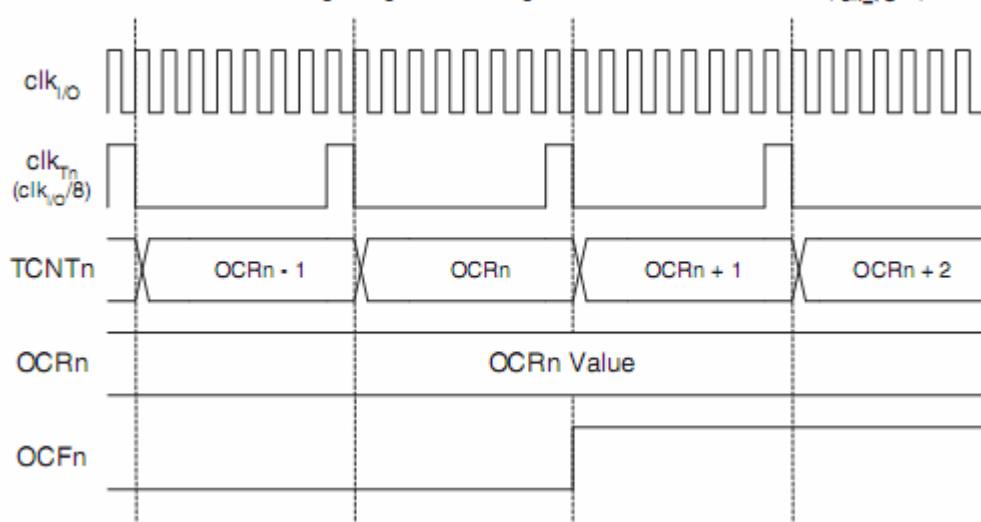
**Figure 41. Timer/Counter Timing Diagram, No Prescaling**



Hình 42 chỉ ra các dữ liệu thời gian giống nhau , nhưng với các bộ đếm gộp trước đã kích hoạt

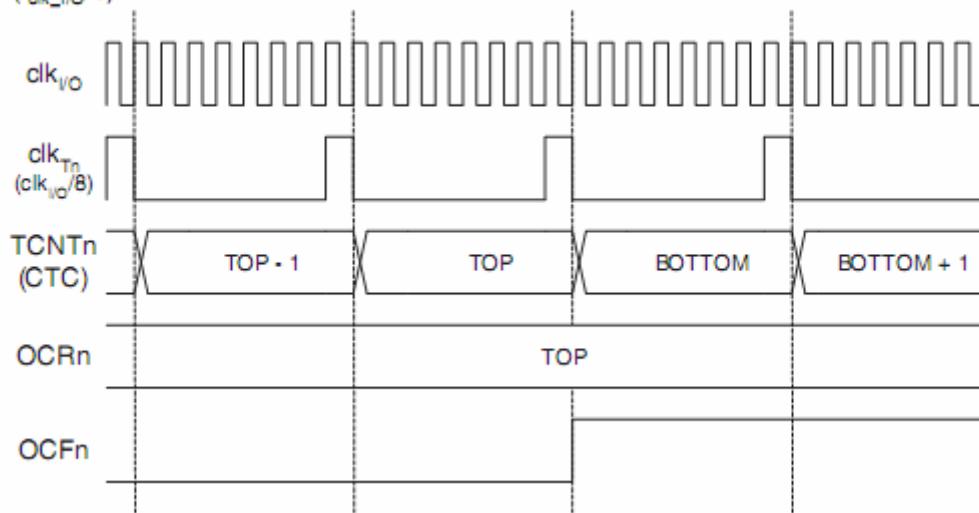
**Figure 42.** Timer/Counter Timing Diagram, with Prescaler ( $f_{clk_{IO}}/8$ )

Hình 43 chỉ ra cách cài đặt OCF0 trong tất cả các chế độ trừ chế độ CTC

**Figure 43.** Timer/Counter Timing Diagram, Setting of OCF0, with Prescaler ( $f_{clk_{IO}}/8$ )

Hình 44 chỉ ra cách cài đặt của OCF0 và xóa TCNT0 trong chế độ CTC

**Figure 44.** Timer/Counter Timing Diagram, Clear Timer on Compare Match Mode, with Prescaler ( $f_{clk_{IO}}/8$ )



## Sự miêu tả các thanh ghi của Timer/Counter 8 bit – 8bit Timer/Counter Register Description

Thanh ghi điều khiển Timer/Counter – TCCR0

Bit	7	6	5	4	3	2	1	0	TCCR0
Read/Write	W	R/W							
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – FOC0 : Force Output Compare ( so sánh đầu ra cưỡng bức )

Bit FOC0 chỉ được hoạt động khi mà các bit WGM xác định chế độ không phải là PWM . Tuy nhiên , để đảm bảo tính tương thích với các thiết bị trong tương lai , bit này phải được cài là 0 khi TCCR0 được viết khi hoạt động trong chế độ PWM . Khi việc viết một mức logic 1 lên bit FOC0 , 1 ghép so sánh trung gian bị cưỡng bức trên bộ phận phát dạng sóng . Đầu ra OC0 bị thay đổi theo việc cài đặt các bit COM01:0 của nó . Chú ý rằng bit FOC0 được cài đặt như là 1 strobe . Do vậy nó là 1 giá trị đưa ra trong các bit COM01:0 xác định rõ hiệu lực của việc so sánh cưỡng bức .

1 sự phân tích FOC0 sẽ không sinh ra bất cứ ngắt nào , hoặc nó sẽ xóa các timer trong chế độ CTC sử dụng OCR0 như là TOP .

Bit FOC0 thì luôn được đọc như là 0

Bit 6, 3 – WGM01:0 : chế độ phát sinh dạng sóng

Các bit này điều khiển các chuỗi đếm của bộ đếm , nguồn cho giá trị đếm cực đại (TOP), và loại nào của sự sinh ra các dạng sóng được sử dụng . Các chế độ điều khiển được hỗ trợ bởi Timer/Counter là : Normal mode , Clear Timer on Compare match (CTC) mode , và 2 loại của chế độ điều chế độ rộng xung PWM .Xem thêm bảng 52 và “modes of Operation “ ở trang 98

**Table 52.** Waveform Generation Mode Bit Description

Mode	WGM01 <sup>(1)</sup> (CTC0)	WGM00 <sup>(1)</sup> (PWM0)	Timer/Counter Mode of Operation	TOP	Update of OCR0 at	TOV0 Flag Set on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR0	Immediate	MAX
3	1	1	Fast PWM	0xFF	BOTTOM	MAX

Note: 1. The CTC0 and PWM0 bit definition names are now obsolete. Use the WGM01:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

Bit 5,4 - COM01:0 : chế độ đầu ra ghép so sánh

Các bit này điều khiển việc xử lí các chân so sánh đầu ra (OC0) . Nếu 1 hoặc cả hai trong các bit COM01:0 được cài đặt , đầu ra OC0 ghi đè lên cổng chức năng thông thường của chân I/O được kết nối với nó . Tuy nhiên , chú ý rằng bit thanh ghi định hướng dữ liệu (DDR ) tương ứng với chân OC0 phải được cài đặt để mà kích hoạt bộ điều khiển đầu ra

Khi mà OC0 được kết nối tới chân , chức năng của các bit COM01:0 phụ thuộc vào việc cài đặt các bit WGM01:0 . Bảng 53 chỉ ra các bit chức năng COM01:0 khi mà các bit WGM01:0 được cài đặt lên một chế độ normal hoặc chế độ CTC mode ( không phải PWM)

**Table 53.** Compare Output Mode, non-PWM Mode

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Toggle OC0 on compare match
1	0	Clear OC0 on compare match
1	1	Set OC0 on compare match

Bảng 54 chỉ ra các bit chức năng COM01:0 khi mà các bit WGM01:0 được cài đặt trong chế độ fast PWM

**Table 54.** Compare Output Mode, Fast PWM Mode<sup>(1)</sup>

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match, set OC0 at BOTTOM, (non-inverting mode)
1	1	Set OC0 on compare match, clear OC0 at BOTTOM, (inverting mode)

Note: 1. A special case occurs when OCR0 equals TOP and COM01 is set. In this case, the compare match is ignored, but the set or clear is done at BOTTOM. See "Fast PWM Mode" on page 99 for more details.

Bảng 55 chỉ ra bit chức năng COM01:0 khi mà WGM01:0 được cài đặt trong chế độ PWM đúng pha

**Table 55.** Compare Output Mode, Phase Correct PWM Mode<sup>(1)</sup>

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match when up-counting. Set OC0 on compare match when downcounting.
1	1	Set OC0 on compare match when up-counting. Clear OC0 on compare match when downcounting.

Note: 1. A special case occurs when OCR0 equals TOP and COM01 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 101 for more details.

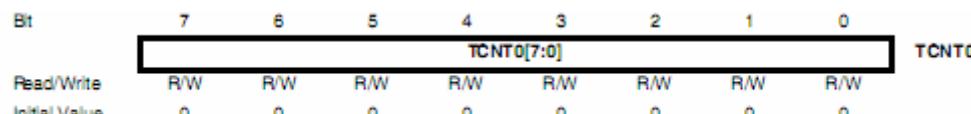
Bit 2:0 : lựa chọn chế độ khóa

Các bit lựa chọn chế độ khóa nguồn khóa được sử dụng bởi Timer/Counter xem bảng 56

**Table 56.** Clock Select Bit Description

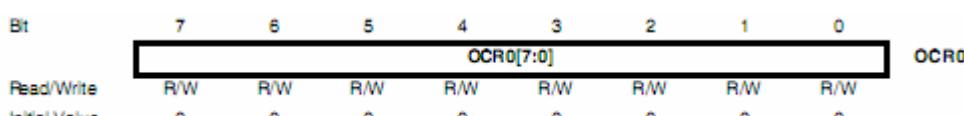
CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	clk <sub>TOS</sub> /(No prescaling)
0	1	0	clk <sub>TOS</sub> /8 (From prescaler)
0	1	1	clk <sub>TOS</sub> /32 (From prescaler)
1	0	0	clk <sub>TOS</sub> /64 (From prescaler)
1	0	1	clk <sub>TOS</sub> /128 (From prescaler)
1	1	0	clk <sub>TOS</sub> /256 (From prescaler)
1	1	1	clk <sub>TOS</sub> /1024 (From prescaler)

### Thanh ghi Timer/Counter - TCNT0



Thanh ghi Timer/Counter đưa ra sự truy nhập trực tiếp , cả hai quá trình điều khiển đọc và ghi , lên bộ đếm 8bit của Timer/Counter . Việc viết lên các khóa thanh ghi (gỡ bỏ ) ghép so sánh trên các xung nhịp timer dưới đây . Sự thay đổi của bộ đếm (TCNT0) trong khi bộ đếm đang chạy , đưa ra 1 nguy cơ của việc lỗi 1 ghép so sánh giữa TCNT0 và thanh ghi OCR0

### Thanh ghi so sánh đầu ra - OCR0



Thanh ghi so sánh đầu ra bao gồm 1 giá trị 8 bit mà có thể tiếp tục được so sánh với giá trị của bộ đếm ( TCNT0) . Một sự ghép có thể được sử dụng để sinh ra 1 ngắt so sánh đầu ra , hoặc sinh ra 1 đầu ra dạng sóng trên chân OC0

## Sự hoạt động không đồng bộ của Timer/Counter

### Thanh ghi trạng thái không đồng bộ - ASSR

Bit	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R	R	R	ASSR
Initial Value	0	0	0	0	0	0	0	0	

Bit 3 – AS0 : Timer/Counter0 dị bộ

Khi AS0 được viết là 0 , Timer/Counter 0 bị khóa khỏi xung nhịp I/O ,  $\text{clk}_{\text{I/O}}$ . Khi AS0 được viết là 1 , Timer/Counter bị khóa khỏi 1 bộ tạo dao động thạch anh được kết nối với chân của bộ tạo dao động timer 1 (TOSC1) . Khi mà giá trị của AS0 bị thay đổi , thành phần của TCNT0 , OCR0 và TCCR0 có thể bị hư hỏng

Bit 2 – TCN0UB : Timer/Counter cập nhật các trạng thái bận (Timer/Counter0 Update Busy )

Khi Timer/Counter0 hoạt động một cách dị bộ và TCNT0 được ghi , bít này được cài đặt . Khi TCNT0 được cập nhật từ thanh ghi lưu trữ tạm thời , bit này được xóa bằng phần cứng . 1 mức logic 0 trong bit này hiển thị cái mà TCNT0 sẵn sàng được cập nhật với 1 giá trị mới .

Bit 1 – OCR0UB : thanh ghi 0 cập nhật trạng thái bận so sánh đầu ra

Khi Timer/Counter 0 hoạt động một cách dị bộ và bit OCR0 được ghi , bit này được cài đặt . Khi OCR0 được cập nhật từ thanh ghi lưu trữ tạm thời , bit này được xóa bằng phần cứng . 1 mức logic 0 trong bit này hiển thị cái mà OCR0 sẵn sàng được cập nhật với 1 giá trị mới .

Bit 0 – TCR0UB Thanh ghi0 điều khiển Timer/Counter cập nhật trạng thái bận

Khi Timer/Counter 0 hoạt động một cách dị bộ và bit TCCR0 được ghi , bit này được cài đặt . Khi TCCR0 được cập nhật từ thanh ghi lưu trữ tạm thời , bit này được xóa bằng phần cứng . 1 mức logic 0 trong bit này hiển thị cái mà TCCR0 sẵn sàng được cập nhật với 1 giá trị mới

Nếu 1 quá trình ghi được tiến hành lên bất cứ cái nào trong 3 thanh ghi của Timer/Counter0 trong khi cờ cập nhật chế độ bận được cài đặt , giá trị cập nhật có thể có hư hỏng và gây ra việc 1 ngắt vô tình xuất hiện .

Bộ phận cho việc đọc TCNT0 , OCR0 và TCCR0 thì khác nhau . Khi việc đọc TCNT0, giá trị timer thực được đọc . Khi đọc OCR0 hoặc TCCR0 , giá trị trong thanh ghi lưu trữ tạm thời được đọc

### Quá trình điều khiển dị bộ của Timer/Counter0

Khi Timer/Counter 0 hoạt động dị bộ , 1 vài sự chú ý đến phải được xem xét

- Cảnh báo : khi chuyển mạch giữa các bộ định thời đồng bộ và không đồng bộ của Timer/Counter0 , thanh ghi Timer TCNT0 , OCR0 và TCCR0 có thể hư hỏng . Một quy trình an toàn cho việc chuyển mạch các nguồn xung nhịp là :
  1. Vô hiệu hóa các ngắt của Timer/Counter0 bằng việc xóa OCIE0 và TOIE0

2. lựa chọn nguồn phát xung nhịp bằng việc cài đặt AS0 như được phê chuẩn
  3. viết giá trị mới lên TCNT0 , OCR0 , và TCCR0
  4. để chuyển mạch sang quá trình điều khiển dị bộ : đợi TCN0UB , OCR0UB , TCR0UB
  5. xóa các cờ ngắt của Timer/Counter0
  6. Kích hoạt các ngắt nếu thấy cần thiết
- Bộ tạo dao động thì được tối ưu cho việc sử dụng với đồng hồ thạch anh 32.768 kHz . Việc áp dụng một xung nhịp ngoài lên chân TOSC1 có thể có kết quả sai trong quá trình điều khiển Timer/Counter 0 . Tần số Xung nhịp chính của CPU phải lớn hơn 4 lần tần số của bộ tạo dao động
  - Khi viết tới 1 trong số các thanh ghi TCNT0 , OCR0 và TCCR0 , giá trị được chuyển vào trong 1 thanh ghi tạm thời , và được chốt sau 2 sườn dương trên TOSC1 . Người sử dụng không nên viết 1 giá trị mới trước khi dung lượng của thanh ghi tạm thời được chuyển đến đích đến của nó . Mỗi 1 thanh ghi trong 3 thanh ghi được nói đến đều có thanh ghi riêng biệt của chúng , điều này có nghĩa là ví dụ : việc viết lên TCNT0 không gây nhiễu 1 quá trình ghi trong OCR0 đang được tiến hành . Để dò 1 sự chuyển đến thanh ghi đích đã xảy ra hay chưa , thì thanh ghi trạng thái dị bộ - ASSR phải được cài đặt .
  - Khi truy nhập vào chế độ Power-save hoặc chế độ Extended Standby sau khi đang được viết lên TCNT0 , OCR0 hoặc TCCR0 , người sử dụng phải đợi cho đến khi thanh ghi được viết được cập nhật nếu Timer/Counter0 được sử dụng để đánh thức thiết bị . Nói cách khác MCU sẽ truy nhập vào chế độ ngủ trước khi các sự thay đổi có hiệu lực . Điều này thì đặc biệt quan trọng nếu như bộ ngắt so sánh đầu ra 0 được sử dụng để đánh thức thiết bị này , từ khi chức năng so sánh đầu ra được vô hiệu hóa trong suốt quá trình viết lên OCR0 hoặc TCNT0 . Nếu chu kì viết không hoàn thành , và MCU truy nhập vào chế độ ngủ trước khi bit OCROUB được trả lại là 0 , thiết bị này sẽ không bao giờ nhận ngắt ghép so sánh , và MCU sẽ không được đánh thức.
  - Nếu Timer/Counter0 được sử dụng để đánh thức thiết bị từ chế độ Power-save hoặc chế độ Extended Standby , 1 sự phòng ngừa phải được thực hiện nếu người sử dụng muốn truy nhập lại vào 1 trong các chế độ này : Mức logic ngắt cần thiết1 chu kì xung TOSC1 để Reset . Nếu thời gian giữa quá trình đánh thức và truy nhập lại các chế độ ngủ nhỏ hơn 1 chu kì TOSC1 , ngắt sẽ không xuất hiện và thiết bị sẽ bị lỗi khởi động . Nếu người sử dụng nghi ngờ 1 trong 2 khoảng thời gian trước khi đăng nhập lại vào chế độ Power-save hoặc chế độ Extended Standby là chế độ ổn định hay không, thuật toán dưới đây có thể được sử dụng để đảm bảo rằng 1 chu kì TOSC1 có kết thúc .
    1. Viết 1 giá trị lên TCCR0 , TCNT0 hoặc OCR0
    2. đợi cho đến khi cờ cập nhật trạng thái bạn tương ứng trong thanh ghi ASSR là 0
    3. truy nhập vào chế độ Power-save hoặc chế độ Extended Standby

- Khi mà quá trình điều khiển dị bộ được lựa chọn , bộ tạo dao động 32.768 kHz cho Timer/Counter0 luôn đang chạy , ngoại trừ trong chế độ Power-down hoặc chế độ Standby . Sau khi 1 sự Reset bật nguồn hoặc đánh thức từ chế độ Power-down hoặc chế độ Standby , người sử dụng nên nhận biết 1 cách chính xác rằng bộ tạo dao động có thể lây đi chỉ cần 1s để ổn định . Người sử dụng thì được khuyên đợi trong khoảng thời gian nhỏ hơn 1s trước khi sử dụng Timer/Counter 0 sau khi bật nguồn hoặc đánh thức từ chế độ ngắt nguồn hoặc chế độ chờ . Dung lượng của tất cả các thanh ghi phải được xét đến tổn thất trước sau khi được đánh thức từ chế độ ngắt nguồn hoặc chế độ chờ xứng với tín hiệu xung nhịp không ổn định trên start –up . Thực chất chỉ một trong 2 thứ là bộ tạo dao động được sử dụng hoặc tín hiệu xung nhịp đồng hồ được áp dụng cho chân TOSC1
- Sự miêu tả của việc đánh thức từ chế độ Power-save hoặc chế độ Extended Standby khi được khóa dị bộ : khi điều kiện ngắt được phù hợp , tiến trình đánh thức được khởi động trên các chu kì của xung nhịp timer bên dưới , cái mà Timer luôn được hoàn thiện bởi dưới 1s trước khi bộ xử lý có thể đọc các giá trị của bộ đếm . Sau khi đánh thức , MCU bị dừng trong 4 chu kì xung nhịp , nó thực thi các chương trình con phục vụ ngắt , và nối tiếp các sự thực thi từ các lệnh dưới lệnh SLEEP
- Quá trình đọc của thanh ghi TCNT0 ngắn sau khi được đánh thức từ chế độ tiết kiệm nguồn có thể đưa ra 1 kết quả không chính xác . Từ khi TCNT0 bị khóa trên khóa không đồng bộ TOSC , việc đọc TCNT0 phải được thực hiện thông qua 1 thanh ghi được đồng bộ hóa tới các vùng xung nhịp I/O bên trong .

Quá trình đồng bộ phải được xảy ra cho mỗi sùơn lên của TOSC1 . Khi mà sự đánh thức khởi chế độ Power-save , và các xung nhịp I/O hoạt động trở lại , TCNT0 sẽ được đọc như là giá trị trước( trước khi truy nhập vào chế độ ngủ ) cho đến khi có sùòn lên tiếp theo của TOSC1 . Pha của xung nhịp TOSC sau quá trình đánh thức từ chế độ tiết kiệm nguồn thì không thể tiên đoán trước một cách cơ bản , như nó phụ thuộc vào thời gian đánh thức . Quy trình được đề nghị cho việc đọc TCNT0 thì được chỉ ra dưới đây :

1. Viết bất cứ giá trị nào tới 1 trong 2 thanh ghi OCR0 hoặc TCCR0
  2. đợi cho cờ cập nhật trạng thái bộ phận tương ứng được xóa
  3. Đọc TCNT0
- trong suốt quá trình điều khiển dị bộ, sự đồng bộ hoá của các cờ ngắt trong khoảng thời gian dị bộ tạo ra ba chu kì xung nhịp của bộ vi xử lý nhiều hơn trong 1 chu kì . Bộ định thời do vậy được nâng cao bởi một bộ phận trước khi bộ vi xử lý có thể đọc giá trị thời gian của việc cài đặt các cờ ngắt. Các chân so sánh công ra được thay đổi trên các bộ định thời và không được đồng bộ hóa tới các xung nhịp của bộ vi xử lý
  -

## thanh ghi che các ngắt của Timer/Counter – TIMSK

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	TIMSK							
Initial Value	0	0	0	0	0	0	0	0	

Bit 1 – OCIE0 : kích hoạt các ngắt ghép so sánh đầu ra Timer/Counter0

Khi mà bit OCIE0 được viết là 1 , và bit I trong thanh ghi trạng thái được đặt là 1 , ngắt ghép so sánh Timer/Counter 0 được kích hoạt . Ngắt tương ứng được thực thi nếu 1 ghép so sánh trong Timer/Couter0 xuất hiện ví dụ như khi bit OCF0 được cài đặt trong thanh ghi cờ báo ngắt Timer/Counter – TIFR

Bit 0 – TOIE0 : kích hoạt ngắt tràn Timer/Couter0

Khi mà bit TOIE0 được viết là 1 , và bit I trong thanh ghi trạng thái được cài đặt là 1 , ngắt dòng tràn Timer/Couter0 được kích hoạt ví dụ như khi bit TOV0 được cài đặt trong thanh ghi cờ báo ngắt – TIFR

## Thanh ghi cờ báo ngắt Timer/Counter –TIFR

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	TIFR							
Initial Value	0	0	0	0	0	0	0	0	

Bit 1 – OCF0 : cờ báo so sánh đầu ra 0

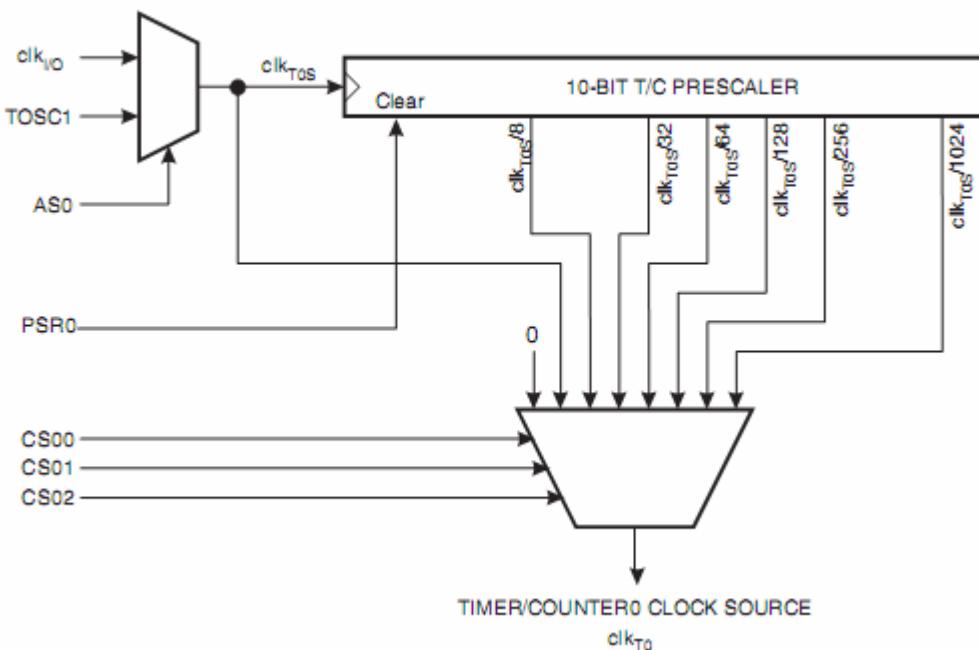
Bit OCF0 được cài đặt là 1 khi 1 ghép so sánh xuất hiện giữa Timer/Counter 0 và dữ liệu ở trong OCR0 – Output Compare Register 0 . OCF0 bị xóa bằng phần cứng khi mà sự thực thi các vec to điều khiển ngắt tương ứng. Như một sự lựa chọn , OCF0 bị xóa bằng việc viết 1 mức logic 1 lên cờ . Khi mà bit I trong SREG, OCIE0 (kích hoạt ngắt ghép so sánh Timer/Counter 0 ) , và OCF0 được cài đặt là 1 , ngắt ghép so sánh Timer/Counter 0 được thực thi

Bit 0 – TOV0 : cờ báo tràn Timer/Counter0

Bit TOV0 được cài đặt là 1 khi 1 dòng tràn xuất hiện trong Timer/Counter 0. TOV0 bị xóa bằng phần cứng khi mà việc thực thi các vec to điều khiển các ngắt tương ứng . Như một sự lựa chọn , TOV0 bị xóa bằng việc viết 1 mức logic 1 lên cờ . Khi mà Bit I SREG ,TOIE0 (kích hoạt cờ báo ngắt tràn Timer/Counter ), và TOV0 được cài đặt là 1 , ngắt tràn Timer/Counter 0 được thực thi . Trong chế độ PWM , bit này được cài đặt khi Timer/Counter 0 thay đổi hướng đếm tại \$00

## Bộ đếm gộp trước Timer/Counter – Timer/Counter Prescaler

Figure 45. Prescaler for Timer/Counter0



Nguồn phát xung nhịp cho Timer/Counter0 được đặt tên là  $\text{clk}_{T0}$  .  $\text{clk}_{T0}$  thì được mặc định kết nối tới xung nhịp chính của hệ thống  $\text{clk}_{I/O}$  . Bằng việc cài đặt bit AS0 trong ASSR , Timer/Counter0 được khóa 1 cách dị bộ từ chân TOSC1 . Điều này kích hoạt sử dụng của Timer/Counter0 như là một bộ đếm thời gian thực (Real Time Counter – RTC ) . Khi bit AS0 được cài đặt , các chân TOSC1 và TOSC2 được ngắt kết nối khỏi cổng C . Một bộ tạo dao động thạch anh có thể sau đó được kết nối giữa các chân TOSC1 và TOSC2 để phục vụ như là 1 nguồn phát xung nhịp độc lập cho Timer/Counter0 . Bộ tạo dao động thì được tối ưu hóa để sử dụng với bộ tạo dao động thạch anh 32.768 kHz . Việc áp dụng một nguồn phát xung nhịp ngoài lên TOSC1 thì không được khuyến khích .

Về Timer/Counter0 , sự lựa chọn tỉ lệ là có thể :  $\text{clk}_{TOS}/8$  ,  $\text{clk}_{TOS}/32$  ,  $\text{clk}_{TOS}/64$  ,  $\text{clk}_{TOS}/128$  ,  $\text{clk}_{TOS}/256$  ,  $\text{clk}_{TOS}/1024$  .Thêm nữa ,  $\text{clk}_{TOS}$  cũng như 0 (stop) có thể được lựa chọn . Việc cài đặt bit PSR0 trong SFIOR khởi động lại bộ đếm gộp trước . Điều này cho phép người sử dụng để điều khiển với 1 bộ đếm gộp trước có thể tiên đoán trước

## Thanh ghi I/O chức năng đặc biệt - Special Function I/O Register – SFIOR

Bit	7	6	5	4	3	2	1	0	SFIOR
Read/Write	TSM	-	-	-	ACME	PUD	PSR0	PSR321	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – TSM : chế độ đồng bộ hóa Timer/Counter

Việc viết bit TSM lên 1 kích hoạt chế độ đồng bộ hóa Timer/Counter . Trong chế độ này , giá trị mà được viết lên PSR0 và PSR321 thì được giữ , dù cho việc giữ các bộ đếm gộp trước tương ứng sẽ reset các tín hiệu được xác nhận . Điều này cũng đảm bảo rằng các bộ Timer/Counter tương ứng được dừng và có thể được cấu hình tới các giá trị giống nhau mà không có nguy cơ hỏng của một trong số chúng trong suốt quá trình cấu hình . Khi bit TSM được viết là 0 , các bit PSR0 và PSR321 bị xóa bởi phần cứng , và Timer/Counter bắt đầu quá trình đếm 1 cách liên tục.

Bit 1 – PSR0 : bộ đếm gộp trước reset Timer/Counter0

Khi bit này là 1 , bộ đếm gộp trước của Timer/Counter sẽ được reset . Bit này thường được xóa ngay lập tức bằng phần cứng . Nếu bit này được viết trong khi Timer/Counter đang hoạt động trong chế độ dị bộ , bit này sẽ còn lại 1 cho đến khi bộ đếm gộp trước được reset xong . bit này sẽ không được xóa bằng phần mềm nếu bit TSM được cài đặt .

## XII . Timer/Counter 16 bit (Timer/Counter 1 và Timer/Counter 3)

Bộ phận Timer/Counter 16 bit cho phép định thời gian các quá trình thực thi chương trình một cách chính xác (quản lý sự kiện) , phát sinh ra các sóng , và đo thời gian tín hiệu .Các đặc điểm chính là

- Thiết kế 16 bit thật (ví dụ ,cho phép 16bit PWM )
- 3 bộ phận so sánh đầu ra độc lập
- Các thanh ghi so sánh đầu ra được ghi vào bộ đếm kép
- Bộ phận bắt đầu vào
- Khóa cắt nhiễu đầu vào
- Xóa timer trên ghép so sánh (tự động tải lại )
- Không có nhiễu sọc ngang , điều chế độ rộng xung đúng pha ( PWM )
- Chu kỳ PWM thay đổi
- Máy phát tần số
- Bộ đếm sự kiện bên ngoài
- 10 nguồn ngắt độc lập (TOV1 , OCF1A , OCF1B , OCF1C , ICF1 , TOV3 , OCF3A , OCF3B , OCF3C , ICF3 )

Sự hạn chế trong chế độ tương thích với Atmega 103

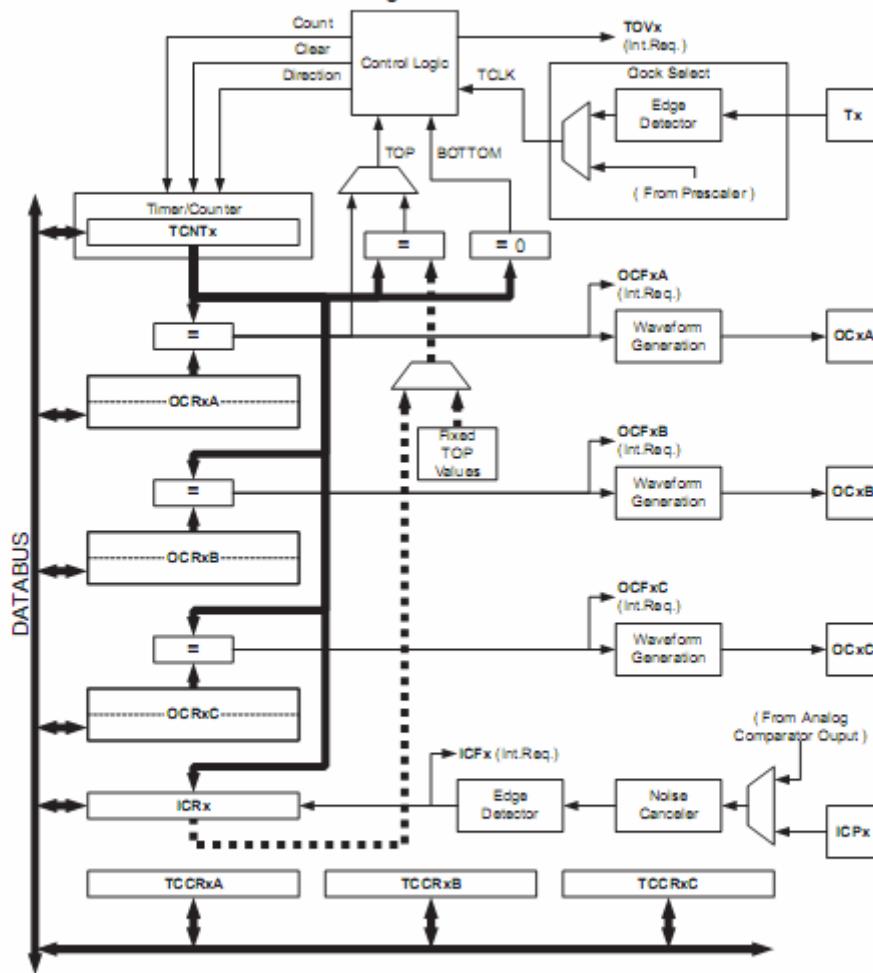
Chú ý rằng trong chế độ tương thích với Atmega 103 , chỉ có 1 Timer/Counter 16 bit là khả dụng(Timer/Counter 1) . Cũng chú ý thêm rằng trong chế độ tương thích với Atmega 103 , Timer/Counter 1 có hai thanh ghi so sánh( compare A và compare B )

### Tổng quan

Tất cả các thanh ghi và các bit tham khảo trong phần này được viết theo dạng chung . 1 trường hợp thấp hơn “n” thay thế cho thứ tự của Timer/Counter , và một trường hợp thấp hơn nữa “x” thay thế cho kênh bộ phận so sánh đầu ra . Tuy nhiên , khi sử dụng thanh ghi hoặc bit xác định trong một chương trình , một mẫu chính xác phải được sử dụng ví dụ như TCNT1 cho việc truy nhập vào giá trị bộ đếm Timer/Counter 1.

Một sơ đồ khái đã rút gọn của Timer/Counter 16 bit được chỉ ra trong hình 46. Về vị trí thật sự của các chân I/O tham khảo phần “Pin Configurations” ở trang 2 . CPU có thể truy nhập vào các thanh ghi I/O, bao gồm các bit I/O , được in đậm . Thanh ghi I/O xác định thiết bị và các bit vị trí được liệt kê trong “16bit Timer/Counter Register Description” ở trang 133.

Figure 46. 16-bit Timer/Counter Block Diagram



Note: Refer to Figure 1 on page 2, Table 30 on page 74, and Table 39 on page 81 for Timer/Counter1 and 3 pin placement and description.

## Các thanh ghi

Timer/Counter (TCNT<sub>n</sub>), thanh ghi so sánh đầu ra (OCR<sub>nA/B/C</sub>), và thanh ghi bắt tín hiệu đầu vào (ICR<sub>n</sub>) tất cả đều là các thanh ghi 16 bit . Các quy trình đặc biệt phải được tuân theo khi truy nhập vào các thanh ghi 16bit. Các quy trình này được miêu tả trong phần “Accessing 16bit Register” ở trang 115 . Các thanh ghi điều khiển Timer/Counter (TCCR<sub>nA/B/C</sub>) là các thanh ghi 8 bit và không có sự hạn chế đối với quyền truy nhập của CPU . Các tín hiệu yêu cầu ngắn (viết tắt Int.Req.) đều được nhìn thấy trong thanh ghi cờ báo ngắn Timer (TIFR) và thanh ghi cờ báo ngắn Timer mở rộng (ETIFR). Tất cả các ngắn đều được che riêng biệt với thanh ghi che ngắn Timer (TIMSK) và thanh ghi che ngắn Timer mở rộng (ETIMSK) . (E)TIFR và (E)TIMSK đều không được chỉ ra trong hình vẽ từ khi các thanh ghi này được chia sẻ bởi các bộ phận Timer khác .

Timer/Counter có thể bị khóa bên trong , bằng bộ đếm gộp trước hoặc bằng 1 nguồn xung nhịp ngoài trên chân Tn. Khối logic lựa chọn xung nhịp điều khiển cái mà nguồn phát xung nhịp và sườn của Timer/Counter sử dụng để làm tăng (hoặc giảm )

giá trị của nó . Timer/Counter thì không hoạt động khi không có nguồn phát xung nhịp nào được lựa chọn . Đầu ra từ logic lựa chọn xung nhịp được tham khảo như là 1 xung nhịp Timer ( $\text{clk}_{\text{Tn}}$ )

Các thanh ghi so sánh đầu ra được ghi vào bộ đếm kép (OCRnA/B/C) thì được so sánh với giá trị của Timer/Counter ở tất cả các thời điểm . Kết quả của việc so sánh có thể được sử dụng bởi máy phát dạng sóng để phát ra 1 PWM hoặc là 1 đầu ra tần số thay đổi trên chân so sánh đầu ra (OcnA/B/C)

Xem thêm phần “Output Compare Units “ ở trang 121 . sự kiện ghép so sánh cũng sẽ đặt cờ ghép so sánh (OCFnA/B/C ) cái mà có thể được sử dụng để sinh ra 1 yêu cầu ngắt so sánh đầu ra .

Thanh ghi bắt đầu vào có thể bắt giá trị của Timer/Counter tại 1 vị trí được đưa ra bên ngoài (sùm khởi động ) sự kiện trên chân bắt tín hiệu cồng ra (ICPn) hoặc trên các chân của bộ so sánh tương tự (xem “Analog Comparator”trên trang 227 . Bộ phận bắt tín hiệu đầu vào bao gồm 1 bộ lọc số ( bộ cắt nhiễu ) cho việc giảm các thay đổi của việc bắt các đỉnh nhiễu .

Giá trị TOP , hoặc giá trị cực đại Timer/Counter , có thể trong 1 vài chế độ của quá trình điều khiển được xác định bằng thanh ghi OCRnA ,hoặc ICRn , hoặc bằng việc cài đặt một giá trị ổn định . Khi việc sử dụng OCRnA như là giá trị TOP trong chế độ PWM , thanh ghi OCRnA không thể được sử dụng cho việc phát sinh ra 1 đầu ra PWM . Tuy nhiên , giá trị TOP ở trong trường hợp này sẽ được ghi vào bộ đếm cho phép giá trị TOP được thay đổi trong thời gian chạy . Nếu như giá trị TOP ổn định là cần thiết , thanh ghi ICRn có thể được sử dụng như là một sự luân phiên , sự tự do của thanhnh OCRnA được sử dụng như là đầu ra PWM .

## Các định nghĩa

Các định nghĩa dưới đây được sử dụng 1 cách rộng rãi xuyên suốt tài liệu này .

**Table 57. Definitions**

BOTTOM	The counter reaches the BOTTOM when it becomes 0x0000.
MAX	The counter reaches its MAXimum when it becomes 0xFFFF (decimal 65535).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be one of the fixed values: 0x00FF, 0x01FF, or 0x03FF, or to the value stored in the OCRnA or ICRn Register. The assignment is dependent of the mode of operation.

## Sự tương thích

Timer/Counter 16bit có nhiều cập nhật và cải tiến từ các phiên bản trước của Timer/Counter 16bit của AVR . Timer/Counter 16 bit này có thể tương thích đầy đủ với các phiên bản sớm hơn .

- Tất cả các Timer/Counter 16bit đều liên quan đến các vị trí địa chỉ thanh ghi I/O , bao gồm các thanh ghi ngắt timer
- Các vị trí bit bên trong tất cả các thanh ghi Timer/Counter 16bit , bao gồm các thanh ghi ngắt Timer
- Các vec tơ ngắt

Các bit điều khiển dưới đây đã được thay đổi tên , nhưng có cùng chức năng và vị trí các thanh ghi

- PWMn0 được chuyển thành WGMn0
- PWMn1 được chuyển thành WGMn1
- CTCn được chuyển thành WGMn2

Các thanh ghi dưới đây thì được thêm vào Timer/Counter 16bit :

- Thanh ghi điều khiển Timer/Counter C (TCCRnC)
- Thanh ghi so sánh đầu ra C , OCRnCH và OCRnCL , được nối với OCRnC

Các bít dưới đây được thêm vào thanh ghi điều khiển Timer/Counter 16bit

- COM1C1:0 được thêm vào TCCR1A
- FOCnA , FOCnB và FOCnC được thêm vào thanh ghi mới TCCRnC
- WGMn3 được thêm vào TCCRnB

Còn ngắt và các bit che cho bộ phận so sánh đầu ra C được thêm vào

Timer/Counter 16bit có sự hoàn thiện mà sẽ có hiệu lực tương thích trong vài trường hợp đặc biệt

## Các thanh ghi 16 bit truy nhập – Accessing 16bit Registers

TCNTn , OCRnA/B/C , và ICRn đều là các thanh ghi 16bit cái mà có thể được truy nhập bởi CPU AVR thông qua bus dữ liệu 8bit . Thanh ghi 16bit phải được đánh địa chỉ byte sử dụng 2 quá trình đọc và ghi . Mỗi Timer 16bit có 1 thanh ghi đơn 8bit cho việc lưu trữ tạm thời của các byte cao của việc truy nhập 16bit . Thanh ghi tạm thời giống nhau được chia sẻ giữa tất cả các thanh ghi 16bit trong mỗi timer 16bit . Sự truy nhập các byte thấp khởi động quá trình đọc và ghi 16bit . Khi byte thấp của thanh ghi 16bit được viết bởi CPU , byte cao được lưu trữ trong thanh ghi tạm thời , và byte thấp đã viết cả hai được sao chép vào trong thanh ghi 16 bit trong chu kì xung nhịp giống nhau . Khi byte thấp của 1 thanh ghi 16bit được đọc bởi CPU , byte cao của thanh ghi 16bit được sao chép vào trong thanh ghi tạm thời trong chu kì xung nhịp giống nhau như lfa byte thấp được đọc

Không phải tất cả các sự truy nhập 16bit sử dụng thanh ghi tạm thời cho byte cao . Việc đọc các thanh ghi OCRnA/B/C thì không bao hàm việc sử dụng thanh ghi tạm thời (Temporary Register)

Để làm việc viết 16bit , byte cao phải được viết trước byte thấp . Vì 1 quá trình đọc 16bit , byte thấp phải được đọc trước byte cao

Đoạn code mẫu dưới đây chỉ ra cách để truy nhập các thanh ghi 16bit Timer giả định rằng không có ngắt nào được cập nhật vào thanh ghi tạm thời . Nguyên tắc giống như vậy có thể được sử dụng 1 cách trực tiếp cho việc truy nhập các thanh ghi

OCRnA/B/C và các thanh ghi ICRn . Chú ý rằng khi sử dụng “C” , trình biên dịch điều khiển sự truy nhập 16bit

Assembly Code Examples <sup>(1)</sup>
<pre> ... ; Set TCNTn to 0x01FF ldi r17,0x01 ldi r16,0xFF out TCNTnH,r17 out TCNTnL,r16 ; Read TCNTn into r17:r16 in r16,TCNTnL in r17,TCNTnH ... </pre>
C Code Examples <sup>(1)</sup>
<pre> unsigned int i; ... /* Set TCNTn to 0x01FF */ TCNTn = 0x01FF; /* Read TCNTn into i */ i = TCNTn; ... </pre>

Note: 1. See "About Code Examples" on page 9.

The assembly code example returns the TCNTn value in the r17:r16 register pair.

Nó thì rất quan trọng để nhận biết rằng việc truy nhập các thanh ghi 16bit là quá trình hoạt động nguyên tử . Nếu như một ngắt xuất hiện giữa 2 lệnh truy nhập thanh ghi 16bit , và mã ngắt cập nhật thanh ghi tạm thời bằng việc truy nhập các thanh ghi Timer giống nhau hoặc bất cứ thanh ghi timer nào khác , sau đó kết quả của việc truy nhập bên ngoài các ngắt sẽ bị hư hỏng . Vì vậy , khi cả hai đoạn mã chính và sự cập nhật các ngắt vào thanh ghi tạm thời, đoạn mã chính phải vô hiệu hóa các ngắt trong suốt sự truy nhập 16bit

Đoạn mã mẫu dưới đây chỉ ra cách để làm 1 quá trình đọc nguyên tử của các thành phần của thanh ghi TCNTn . Việc đọc bất cứ thanh ghi OCRnA/B/C hoặc ICRn có thể thực hiện bằng cách sử dụng các nguyên tắc giống nhau .

**Assembly Code Example<sup>(1)</sup>**

```

TIM16_ReadTCNTn:
    ; Save global interrupt flag
    in r18,SREG
    ; Disable interrupts
    cli
    ; Read TCNTn into r17:r16
    in r16,TCNTnL
    in r17,TCNTnH
    ; Restore global interrupt flag
    out SREG,r18
    ret

```

**C Code Example<sup>(1)</sup>**

```

unsigned int TIM16_ReadTCNTn( void )
{
    unsigned char sreg;
    unsigned int i;
    /* Save global interrupt flag */
    sreg = SREG;
    /* Disable interrupts */
    __disable_interrupt();
    /* Read TCNTn into i */
    i = TCNTn;
    /* Restore global interrupt flag */
    SREG = sreg;
    return i;
}

```

Note: 1. See "About Code Examples" on page 9.

Ví dụ đoạn mã Assembly trả lại giá trị của TCNTn trong cặp thanh ghi r17:r16

Đoạn mã mẫu dưới đây chỉ ra cách để làm 1 quá trình viết nguyên tử của các thành phần của thanh ghi TCNTn . Việc viết bất cứ các thanh ghi OCRnA/B/C hoặc ICRn có thể được thực hiện bằng các nguyên lí giống nhau .

**Assembly Code Example<sup>(1)</sup>**

```

TIM16_WriteTCNTn:
    ; Save global interrupt flag
    in r18,SREG
    ; Disable interrupts
    cli
    ; Set TCNTn to r17:r16
    out TCNTnH,r17
    out TCNTnL,r16
    ; Restore global interrupt flag
    out SREG,r18
    ret

```

**C Code Example<sup>(1)</sup>**

```

void TIM16_WriteTCNTn( unsigned int i )
{
    unsigned char sreg;
    unsigned int i;
    /* Save global interrupt flag */
    sreg = SREG;
    /* Disable interrupts */
    __disable_interrupt();
    /* Set TCNTn to i */
    TCNTn = i;
    /* Restore global interrupt flag */
    SREG = sreg;
}

```

Note: 1. See "About Code Examples" on page 9.

The assembly code example requires that the r17:r16 register pair contains the value to be written to TCNTn.

## Khả năng dùng lại của thanh ghi byte cao tạm thời .

Nếu việc viết vào nhiều hơn 1 thanh ghi 16bit nơi mà các byte cao thì giống nhau cho tất cả các thanh ghi được viết , sau đó các byte cao chỉ cần được viết 1 lần . Tuy nhiên , chú ý rằng nguyên tắc giống nhau của quá trình điều khiển nguyên tử được miêu tả trước đó cũng được áp dụng trong trường hợp này .

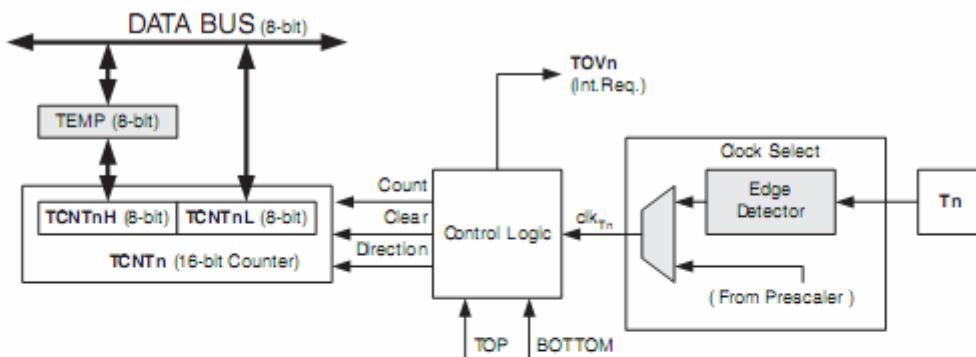
## Các nguồn phát xung nhịp Timer/Counter

Timer/Counter có thể bị khóa bằng nguồn phát xung nhịp bên trong hoặc bên ngoài. Nguồn phát xung nhịp được lựa chọn bằng mức logic lựa chọn xung nhịp cái mà được điều khiển bằng các bit lựa chọn xung nhịp(CSn2:0 ) được đặt trong thanh ghi điều khiển Timer/CounterB (TCCRnB) . Để biết thêm chi tiết trên nguồn phát xung nhịp và các bộ đếm gộp trước xem thêm “ các bộ đếm gộp trước Timer/Counter3 ,Timer/Counter2, và Timer/Counter1” ở trang 144

## Thành phần bộ đếm – Counter Unit

Các phần chính của Timer/Counter16bit thì có thể lập trình thành phần bộ đếm 16 bit 2 hướng . Hình 47 chỉ ra 1 sơ đồ khái của các bộ đếm và các thành phần xung quanh nó .

**Figure 47. Counter Unit Block Diagram**



Signal description (internal signals):

- Count** Increment or decrement TCNTn by 1.
- Direction** Select between increment and decrement.
- Clear** Clear TCNTn (set all bits to zero).
- clk<sub>Tn</sub>** Timer/Counter clock.
- TOP** Signalize that TCNTn has reached maximum value.
- BOTTOM** Signalize that TCNTn has reached minimum value (zero).

Bộ đếm 16bit được vẽ bản đồ vào trong 2 vị trí bộ nhớ I/O 8 bit : Bộ đếm cao (TCNTnH) bao gồm nhiều hơn 8bit của bộ đếm , và bộ đếm thấp (TCNTnL) bao gồm thấp hơn 8bit . Thanh ghi TCNTnH có thể được truy nhập một cách trực tiếp bởi CPU .

Khi mà CPU thực hiện một truy nhập đến địa chỉ I/O TCNTnH , CPU truy nhập byte cao trong thanh ghi tạm thời (TEMP) . Thanh ghi tạm thời thì được cập nhật với giá trị TCNTnH khi mà TCNTnL được đọc , và TNCTnH thì được cập nhật với giá trị của thanh ghi tạm thời khi mà TCNTnL được ghi . Điều này cho phép CPU để đọc hoặc và ghi giá trị trọn vẹn của bộ đếm 16 bit trong vòng 1 chu kì xung nhịp thông qua bus dữ liệu 8 bit . Nó thì thực sự quan trọng để nhận biết rằng có những trường hợp đặc biệt của việc viết lên các thanh ghi TCNTn khi mà bộ đếm đang đếm cái mà sẽ đưa ra những kết quả không thể tiên đoán trước được . Những trường hợp đặc biệt này thì được miêu tả trong các phần quan trọng phía sau .

Phụ thuộc vào các kiểu điều khiển được sử dụng , các bộ đếm bị xóa , được làm tăng , hoặc giảm tại mỗi xung nhịp Timer ( $clk_{Tn}$ ) .  $clk_{Tn}$  có thể được sinh ra từ 1 nguồn phát xung nhịp ngoài hoặc bên trong , được lựa chọn bởi các bit lựa chọn xung nhịp (CSn2:0) . Khi không có nguồn phát xung nhịp nào được lựa chọn (CSn2:0 = 0) bộ timer được dừng lại . Tuy nhiên, giá trị TCNTn có thể được truy nhập bởi CPU, sự độc lập của  $clk_{Tn}$  thì được đưa ra hoặc không . 1 CPU ghi đè (có quyền ưu tiên ở trên ) lên tất cả các bộ đếm xóa hoặc các quá trình đếm .

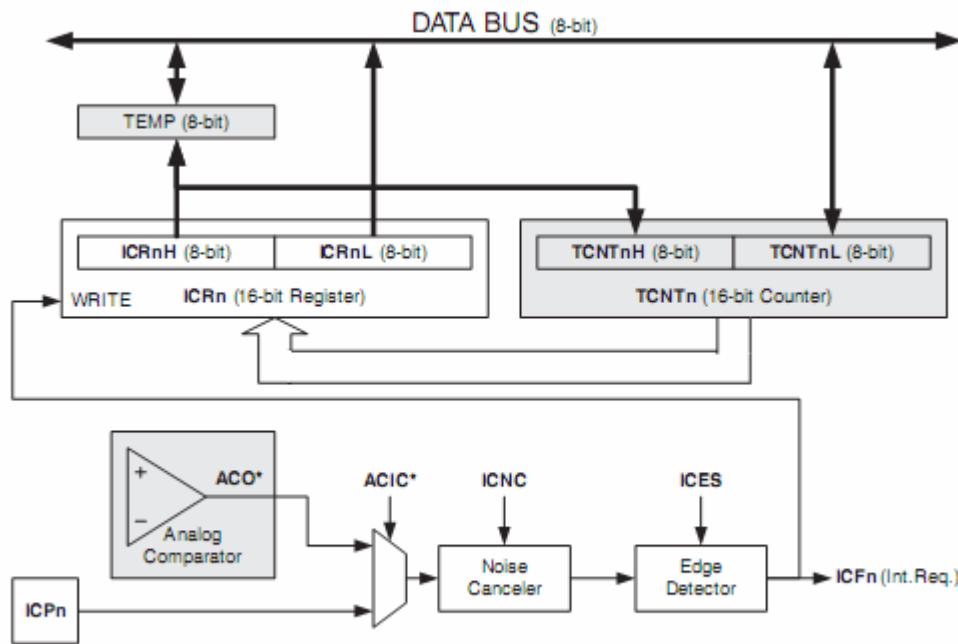
Các chuỗi đếm thì được xác định bằng việc cài đặt của các bit chế độ phát sinh dạng sóng (WGMn3:0) được đặt ở trong thanh ghi điều khiển Timer/Counter A và B (TCCRnA và TCCRnB). Có các kết nối đóng giữa cách các bộ đếm được xử lí (đếm) và cách mà các dạng sóng được phát ra trên các đầu ra so sánh đầu ra Ocnx . Để biết thêm chi tiết về chuỗi đếm hoàn thiện và máy phát dạng sóng , xem thêm bảng “các chế độ điều khiển “ trên trang 124.

Còn báo dòng tràn Timer/Counter (TOVn) thì được cài đặt theo chế độ điều khiển được lựa chọn bằng các bit WGM3:0 . TOVn có thể được sử dụng để phát sinh ra các ngắt CPU.

## Bộ phận bắt tín hiệu đầu ra – Input Capture Unit

Timer/Counter thì không chính xác là 1 bộ phận bắt tín hiệu đầu vào cái mà có thể bắt các sự kiện bên ngoài và đưa chúng vào 1 khuôn thời gian hiển thị thời gian của các sự cố . Tín hiệu bên ngoài hiển thị 1 sự kiện , hoặc nhiều sự kiện , có thể được áp dụng thông qua chân ICPn hoặc 1 cách luân phiên , chỉ Timer/Counter 1 , thông qua bộ phận so sánh tương tự. Khuôn thời gian có thể sau đó được sử dụng để tính toán tần số , chu kì vòng tải duty-cycle , và các tính năng khác của các tín hiệu được áp dụng . Như một sự lựa chọn , khuôn thời gian có thể được sử dụng cho việc tạo ra 1 biểu đồ của các sự kiện .

Bộ phận bắt tín hiệu vào thì được minh họa như được chỉ ra trong hình 48 . Các thành phần của sơ đồ khối cái mà không trực tiếp là 1 phần của bộ phận bắt tín hiệu đầu vào thì được bôi xám . Chữ n nhỏ trong thanh ghi và các tên bit được hiển thị trong số thứ tự của Timer/Counter .

**Figure 48. Input Capture Unit Block Diagram**

**Note:** The Analog Comparator Output (ACO) can only trigger the Timer/Counter1 ICP – not Timer/Counter3.

Khi 1 sự thay đổi của mức logic (1 biến cõi ) xuất hiện trên chân bắt tín hiệu đầu ra (ICPn), như một sự lựa chọn trên đầu ra của bộ so sánh tương tự (ACO) , và thay đổi này chứng thực tới việc cài đặt của các bộ dò sờn xung , 1 capture sẽ được khởi động . Khi 1 bộ bắt tín hiệu được khởi động , giá trị 16 bit của bộ đếm (TCNTn) được viết tới các thanh ghi bắt tín hiệu đầu vào (ICRn) . Các cờ bắt tín hiệu đầu vào (ICFn) được cài đặt ở các xung nhịp hệ thống giống nhau như là các giá trị của TCNTn được sao chép vào trong thanh ghi ICRn . Nếu được kích hoạt(TICIEn = 1 ) cờ báo bắt tín hiệu đầu vào sinh ra 1 ngắt bắt tín hiệu đầu vào (Input Capture interrupt ) . Cờ ICFn sẽ được xóa 1 cách tự động khi mà ngắt được thực thi . Như một sự lựa chọn cờ ICFn có thể bị xóa bằng phần mềm bằng việc viết 1 mức logic 1 lên vị trí bit I/O của nó .

Việc đọc giá trị 16 bit trong thanh ghi bắt tín hiệu đầu vào (ICRn) được thực hiện trước hết bằng việc đọc byte thấp (ICRnL) và sau đó là các byte cao (ICRnH) . Khi byte thấp được đọc , byte cao được sao chép vào trong thanh ghi dữ liệu tạm thời byte cao (TEMP) . Khi mà CPU đọc vùng I/O ICRnH nó sẽ truy nhập thanh ghi TEMP

Thanh ghi ICRn có thể chỉ được viết khi sử dụng 1 chế độ phát dạng sóng cái mà dùng thanh ghi ICRn cho việc xác định giá trị TOP của bộ đếm . Trong trường hợp này, các bit chế độ phát dạng sóng (Waveform Generation mode ) (WGMn3:0) phải được cài đặt trước khi giá trị TOP có thể được viết tới thanh ghi ICRn . Khi việc viết thanh ghi ICRn các byte cao phải được viết lên vùng I/O ICRnH trước khi byte thấp được viết tới ICRnL

Để thêm thông tin về cách truy nhập thanh ghi 16bit tham khảo “Accessing 16bit Registers “ trên trang 115 .

## Nguồn chân bắt tín hiệu đầu vào

Nguồn khởi động chính cho bộ phận bắt tín hiệu đầu vào là chân bắt tín hiệu đầu vào (ICPn) . Timer/Counter1 có thể sử dụng luân phiên đầu ra bộ so sánh tương tự như là 1 nguồn khởi động cho bộ phận bắt tín hiệu đầu vào . Bộ so sánh tương tự thì được lựa chọn như là nguồn khởi động bằng cách cài đặt bit bắt tín hiệu bộ so sánh tương tự (ACIC) trong thanh ghi trạng thái và điều khiển bộ so sánh tương tự (ACSR) . Để nhận biết nguồn khởi động sự thay đổi có thể khởi động 1 capture . Cờ báo bắt tín hiệu đầu ra do vậy phải được xóa sau khi thay đổi .

Cả hai chân bắt tín hiệu đầu vào (ICPn) và đầu vào của đầu ra bộ so sánh tương tự (ACO) thì được lấy mẫu sử dụng các công nghệ giống nhau như chân Tn (hình 59 trang 144) . Bộ dò sườn thì cũng giống nhau . Tuy nhiên , khi mà bộ khóa cắt nhiễu được kích hoạt , thêm vào đó mức logic được chèn vào trước bộ dò sườn xung (edge detector) , cái mà tăng độ trễ bằng 4 chu kì xung nhịp hệ thống . Chú ý rằng đầu vào của khóa cắt nhiễu và bộ dò sườn thì luôn được kích hoạt trừ phi Timer/Counter được cài đặt trong 1 chế độ phát dạng sóng cái mà sử dụng ICRn để xác định giá trị TOP

Một bộ bắt tín hiệu đầu ra có thể được khởi động bằng phần mềm bởi việc điều khiển cổng của chân ICPn

## Khóa cắt nhiễu – Noise Canceler

Một bộ khóa cắt nhiễu nâng cao việc giảm nhiễu bằng việc sử dụng 1 sơ đồ bộ lọc số đơn giản . Đầu vào bộ khóa cắt nhiễu thì được điều chỉnh qua 4 quá trình lấy mẫu , và tất cả 4 lần đó phải bằng nhau để cho việc thay đổi đầu ra cái mà lần lượt được sử dụng bởi bộ dò sườn xung .

Bộ khóa cắt nhiễu thì được kích hoạt bằng việc cài đặt bit khóa cắt nhiễu bắt tín hiệu đầu vào (ICNCFn) trong thanh ghi điều khiển Timer/Counter B (TCCRnB) . Khi kích hoạt khóa cắt nhiễu đưa vào 4 chu kì xung nhịp hệ thống thêm vào của độ trễ từ 1 thay đổi được áp dụng lên đầu vào , để cập nhật thanh ghi ICRn . Khóa cắt nhiễu sử dụng xung nhịp hệ thống và vì vậy nó không bị ảnh hưởng đến bộ đếm gộp trước .

## Việc sử dụng bộ phận bắt tín hiệu đầu vào

Sự thử thách chính của việc sử dụng bộ phận bắt tín hiệu đầu vào là để gán đủ dung lượng của bộ nhớ cho việc điều khiển các biến cố đến . Khoảng thời gian giữa các biến cố thì có tính quyết định . Nếu bộ vi xử lý không phải đọc giá trị bắt được trong thanh ghi ICRn trước khi biến cố tiếp theo xuất hiện , ICRn sẽ được ghi đè với giá trị mới . Trong trường hợp này kết quả của việc truy bắt sẽ bị sai .

Khi sử dụng ngắt bắt tín hiệu đầu vào (Input Capture) , thanh ghi ICRn nên được đọc sớm trong chương trình con phục vụ ngắt nếu có thể . Dù vậy ngắt của bộ truy bắt tín hiệu đầu vào có quyền ưu tiên cao 1 cách tương đối , thời gian đáp ứng ngắt

cực đại thì phụ thuộc vào số lớn nhất của chu kì xung nhịp nó lấy đi để điều khiển bắt cứ 1 yêu cầu ngắt nào khác .

Việc sử dụng bộ phận bắt tín hiệu đầu ra trong bắt cứ một chế độ điều khiển nào khi mà giá trị TOP được thay đổi một cách chủ động trong suốt quá trình hoạt động , thì không được khuyên nghị .

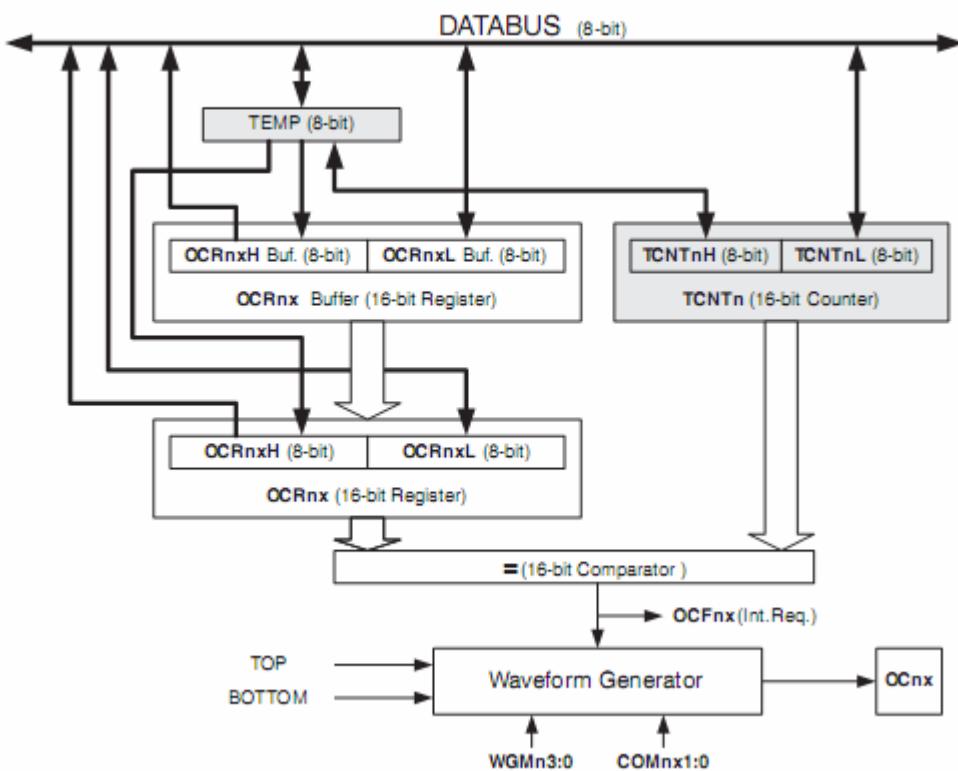
Việc đo của một chu kì tải của 1 tín hiệu bên ngoài cái mà cần thiết để khởi động sùn thì được thay đổi sau mỗi lần truy bắt . Việc thay đổi sự lấy mẫu sùn (Edge sensing) phải được thực hiện sớm nhất có thể sau khi thanh ghi ICRn vừa được đọc . Sau 1 sự thay đổi của sùn xung , cờ báo bắt tín hiệu đầu vào (ICFn) phải được xóa bằng phần mềm (viết mức logic 1 lên vùng địa chỉ bit I/O) . Về việc chỉ đo tần số , quá trình xóa của cờ ICFn thì không cần thiết (nếu một điều khiển ngắt được sử dụng )

## Các bộ phận so sánh đầu ra – Output Compare Units

Bộ so sánh 16 bit tiếp tục so sánh TCNTn với thanh ghi so sánh đầu ra (OCRnx) . Nếu như TCNT bằng OCRnx , các tín hiệu của bộ so sánh có một sự tương ứng . Một match sẽ được cài đặt cờ so sánh đầu ra (OCFnx) tại chu kì xung nhịp tiếp theo . Nếu được kích hoạt (OCIEnx = 1 ) cờ báo so sánh đầu ra sinh ra 1 ngắt so sánh đầu ra . Cờ OCFnx thì tự động được xóa khi ngắt được thực thi . Như một sự lựa chọn , cờ OCFnx có thể bị xóa bằng phần mềm bởi việc viết một mức logic 1 lên vùng địa chỉ bit I/O của nó . Máy phát dạng sóng sử dụng tín hiệu ghép để phát ra 1 tín hiệu đầu ra theo chế độ điều khiển cài đặt bằng các bit chế độ phát dạng sóng (vWGMn3:0) và các bit chế độ đầu ra so sánh (COMnx1:0) . Tín hiệu TOP và BOTTOM được sử dụng bằng máy phát dạng sóng để điều khiển các trường hợp đặc biệt của các giá trị cực biên trong một vài chế độ điều khiển ( Xem bảng “chế độ điều khiển “ trên trang 124 )

Một tính năng đặc biệt của bộ phận đầu ra so sánh cho phép nó xác định giá trị TOP của Timer/Counter (ví dụ độ chính xác của bộ đếm ) .Thêm vào độ chính xác của bộ đếm , giá trị TOP xác định chu kì thời gian cho dạng sóng được phát bằng máy phát dạng sóng .

Hình 49 chỉ ra 1 sơ đồ khối của bộ phận so sánh đầu ra . Chữ “n” nhỏ trong thanh ghi và các tên bit hiển thị số thứ tự của thiết bị (n = n của Timer/Counter n ) , và “x” hiển thị bộ phận đầu ra so sánh (A/B/C) . Thành phần của sơ đồ khối cái mà không trực tiếp là 1 phần của bộ phận đầu ra so sánh thì được bôi xám .

**Figure 49. Output Compare Unit, Block Diagram**

Thanh ghi OCRnx thì được ghi vào bộ đếm kép khi sử dụng bất cứ 1 trong 12 chế độ điều chế độ rộng xung (PWM) . Về chế độ điều khiển bình thường và chế độ so sánh trên xóa timer- Clear Timer on Compare mode (CTC) , sự ghi vào bộ đếm kép thì bị vô hiệu hóa. Sự ghi vào bộ đếm kép thì đồng bộ hóa việc cập nhật của thanh ghi so sánh OCRnx đến 1 trong 2 giá trị của chuỗi đếm là TOP và BOTTOM . Sự đồng bộ hóa ngăn cản sự xuất hiện của odd-length , và các xung PWM không đối xứng , do đó tạo ra các đầu ra không có nhiễu sọc ngang .

Sự truy nhập thanh ghi OCRnx có thể dường như phức tạp , nhưng đó không phải là trường hợp này . Khi bộ đếm kép (double buffering ) được kích hoạt , CPU vừa truy cập lên thanh ghi bộ đếm OCRnx , và nếu bộ đếm kép bị vô hiệu hóa CPU sẽ truy nhập vào OCRnx 1 cách trực tiếp . Thành phần của thanh ghi OCR1x (bộ đếm hoặc bộ so sánh ) chỉ bị thay đổi bằng một quá trình viết (Timer/Counter thì không cập nhật thanh ghi 1 cách tự động như là các thanh ghi TCNTn và ICRn ) . Vì vậy OCRnx thì không được đọc thông qua thanh ghi tạm thời byte cao (TEMP) . Tuy nhiên , nó là 1 sự thực hành tốt để đọc byte thấp trước tiên như khi truy nhập vào các thanh ghi 16bit khác . Quá trình viết các thanh ghi OCRnx phải được thực hiện thông qua thanh ghi TEMP từ khi quá trình so sánh của tất cả các thiết bị 16bit thì được thực hiện 1 cách liên tục . Byte cao (OCRnxH) được ghi trước tiên . Khi vùng địa chỉ I/O byte cao được ghi bởi CPU , thanh ghi TEMP sẽ được cập nhật bởi giá trị đã được viết . Sau đó khi byte thấp (OCRnxL ) được ghi lên các thấp hơn 8bit , byte cao sẽ được sao chép vào trong upper 8bit của 1 trong 2 thanh ghi bộ đếm OCRnx hoặc thanh ghi so sánh OCRnx trong cùng một chu kì xung nhịp hệ thống .

Để thêm thông tin về cách truy nhập vào các thanh ghi 16bit tham khảo thêm “Accessing 16bit Registers “ trên trang 115

### **So sánh đầu ra cưỡng bức**

Trong các chế độ không phải là chế độ phát dạng sóng PWM , đầu ra ghép (match output) của bộ so sánh có thể bị cưỡng bức bởi việc viết một mức logic 1 lên bit so sánh đầu ra cưỡng bức (FOCnx) . Việc ghép so sánh cưỡng bức sẽ không cài đặt cờ OCFnx hoặc reload/clear Timer , nhưng chân Ocnx sẽ được cập nhật như là nếu 1 ghép so sánh xuất hiện( các bit COMnx1:0 cài đặt xác định rằng chân Ocnx được cài đặt , bị xóa hoặc di chuyển )

### **Quá trình khóa ghép so sánh bằng việc viết TCNTn**

Tất cả quá trình viết của CPU lên thanh ghi TCNTn sẽ khóa bất cứ ghép so sánh nào cái mà xuất hiện trong chu kỳ xung nhịp tiếp theo , dù cho khi Timer được dừng lại . Đặc điểm này cho phép OCRnx được khởi tạo lại đến giá trị giống như TCNTn mà không khởi động 1 ngắt khi mà khóa Timer/Counter được kích hoạt .

### **Việc sử dụng bộ phận so sánh đầu ra**

Từ khi việc viết TCNTn trong bất cứ chế độ điều khiển nào sẽ khóa tất cả các ghép so sánh trong 1 chu kỳ xung nhịp timer , có những nguy cơ được kéo theo khi sự thay đổi TCNTn khi sử dụng bất cứ các kênh so sánh đầu ra nào , sự phụ thuộc của việc lựa chọn Timer/Counter đang chạy hay không . Nếu giá trị được viết lên TCNTn bằng giá trị OCRnx , ghép so sánh sẽ bị không đúng , kết quả trong việc phát ra dạng sóng không đúng . Không được viết TCNTn bằng giá trị TOP trong chế độ PWM với các giá trị TOP thay đổi . Ghép so sánh cho TOP sẽ được bỏ qua và bộ đếm sẽ tiếp tục lên 0xFFFF . Một cách tương tự , không được viết giá trị TCNTn bằng giá trị BOTTOM khi mà bộ đếm đang đếm xuống

Sự cài đặt của Ocnx nên được tiến hành trước khi việc cài đặt thanh ghi định hướng dữ liệu cho chân cổng lên cổng ra . Cách dễ nhất để cài đặt giá trị Ocnx là sử dụng các bit phân tích so sánh đầu ra (FOCnx) trong chế độ bình thường (normal). Thanh ghi Ocnx giữ giá trị của nó dù khi thay đổi giữa các chế độ phát dạng sóng

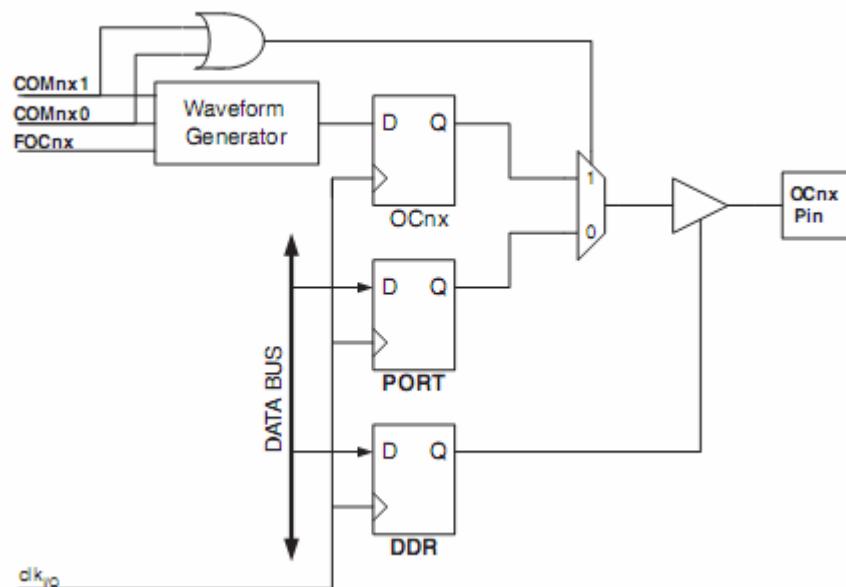
Để nhận ra rằng các bit COMnx1:0 thì không được ghi vào bộ đếm kép cùng nhau với giá trị so sánh . Sự thay đổi các bit COMnx sẽ tạo hiệu lực ngay lập tức .

### **Bộ phận ghép đầu ra so sánh**

Các bit chế độ đầu ra so sánh (COMnx1:0) có hai chức năng . Máy phát dạng sóng sử dụng các bit COMnx1:0 cho việc xác định trạng thái so sánh đầu ra (OCnx) tại ghép so sánh tiếp theo . Thứ hai là các bit COMnx1:0 điều khiển chân OCnx nguồn đầu ra . Hình 50 chỉ ra sơ đồ đã được rút gọn của sự hư hỏng logic bởi việc cài đặt bit

OCMnx 1:0 . Các thanh ghi I/O , các bit I/O và các chân I/O trong hình thì được bôi đậm. Chỉ các phần của các thanh ghi điều khiển cổng I/O chung (DDR và PORT) cái mà bị hỏng bởi các bit COMnx1:0 được chỉ ra . Khi tham khảo đến các trạng thái của OCnx , sự tham chiếu thì cho thanh ghi OCnx bên trong , không phải cho các chân OCnx . Nếu tín hiệu Reset xuất hiện , thanh ghi OCnx được reset về 0

**Figure 50. Compare Match Output Unit, Schematic**



Chức năng của các cổng I/O chung thì được ghi đè bằng bit so sánh đầu ra (OCnx ) từ máy phát dạng sóng nếu 1 trong 2 bit COMnx1:0 được cài đặt . Tuy nhiên , sự định hướng chân OCnx ( đầu vào hoặc đầu ra ) thì vẫn được điều khiển bởi thanh ghi định hướng dữ liệu (DDR) cho chân cổng . Bit thanh ghi định hướng dữ liệu cho chân OCnx (DDR\_OCnx) phải được cài đặt như cổng ra trước khi giá trị của OCnx được nhìn thấy trên chân . Chức năng ghi đè cổng là nguyên tắc độc lập của chế độ phát dạng sóng , nhưng có 1 vài ngoại lệ . Tham khảo bảng 58 bảng 59 và 60 để biết thêm chi tiết .

Thiết kế của chân logic so sánh cổng ra cho phép việc khởi tạo của trạng thái OCnx trước khi cổng ra được kích hoạt . Chú ý rằng sự cài đặt bit COMnx1:0 được dự trữ cho các chế độ đã biết của quá trình điều khiển . Xem “mô tả thanh ghi Timer/Counter 16bit trên trang 133

Các bit COMnx1:0 không có hiệu lực trên bộ phận bắt tín hiệu đầu vào

### Chế độ đầu ra so sánh và sự phát sinh dạng sóng

Máy phát dạng sóng sử dụng các bit COMnx1:0 cách nhau trong chế độ thông thường , và các chế độ PWM . Cho tất cả các chế độ , việc cài đặt các bit COMnx1:0 = 0 nói cho máy phát biết rằng không có hành động nào trên thanh ghi OCnx được tiến hành trong ghép so sánh tiếp theo . Cho các hoạt động đầu ra so sánh trong các chế độ không phải PWM tham khảo bảng 58 trang 133 . Về chế độ fast PWM tham khảo bảng 59 trang 134 , và cho chế độ PWM đúng tần số và PWM đúng pha tham khảo bảng 60 trang 134

Một sự thay đổi trạng thái các bit COMnx1:0 sẽ có hiệu lực tại ghép so sánh đầu tiên sau khi các bit được ghi . Về các chế độ không phải PWM , hoạt động có thể bị cưỡng ép có hiệu lực ngay lập tức bằng việc sử dụng bit phân tích FOCnx

## Các chế độ của quá trình điều khiển

Chế độ của quá trình điều khiển ví dụ như sự xử lí của Timer/Counter và các chân so sánh đầu ra , được xác định bằng việc kết hợp của chế độ phát dạng sóng (WGMn3:0) và các bit đầu ra so sánh (COMnx1:0) . Các bit chế độ đầu ra so sánh thì không ảnh hưởng đến các chuỗi đếm , trong khi các bit chế độ phát dạng sóng thì ảnh hưởng . Các bit COMnx1:0 điều khiển lựa chọn đầu ra PWM được phát ra nên được đảo hay không ( xung PWM đảo hoặc không đảo ) . Cho các chế độ không phải PWM các bit COMx1:0 điều khiển đầu ra lựa chọn nên được cài đặt , bị xóa hay di chuyển tại 1 ghép so sánh ( xem “compare match output unit trên trang 123 )

Để biết thêm chi tiết về thông tin định thời tham khảo “Timer/Counter Timing Diagrams” trên trang 131 .

### Chế độ bình thường – Normal mode

#### Chế độ bình thường

Chế độ đơn giản nhất của quá trình điều khiển là chế độ bình thường (WGMn3:0) = 0. Trong chế độ này việc định hướng đếm thì luôn luôn là lên trên (tăng dần), và không có sự xóa bộ đếm nào được tiến hành. Bộ đếm đơn giản bị tràn khi mà nó vượt qua giá trị cực đại 16 (MAX = 0xFFFF ) và sau đó khởi động lại từ chế độ BOTTOM (0x0000) Trong chế độ điều khiển bình thường cờ báo tràn Timer/Counter (TOVn) sẽ được cài đặt trong cùng chu kì xung nhịp như là TCNTn trở thành 0 . Cờ báo TOVn trong trường hợp này được xử lí như là bit thứ 17 , ngoại lệ là nó chỉ được cài đặt , không bị xóa . Tuy nhiên , được kết hợp với ngắt báo tràn timer cái mà tự động xóa cờ TOVn , độ chính xác của timer có thể được tăng lên bằng phần mềm . Không có trường hợp đặc biệt nào được xét đến trong chế độ bình thường , 1 giá trị bộ đếm mới có thể được ghi bất cứ thời gian nào .

Bộ phận truy bắt tín hiệu đầu ra, thì rất dễ được sử dụng trong chế độ bình thường. Tuy nhiên việc quan sát giá trị cực đại bên trong giữa các biến cố bên ngoài phải không được vượt quá độ chính xác của bộ đếm. Nếu giá trị bên trong của các biến cố quá dài, cờ báo ngắt tràn tham mồi hoặc bộ đếm gộp trước của timer phải được sử dụng để mở rộng độ chính xác của bộ phận truy bắt tín hiệu đầu vào.

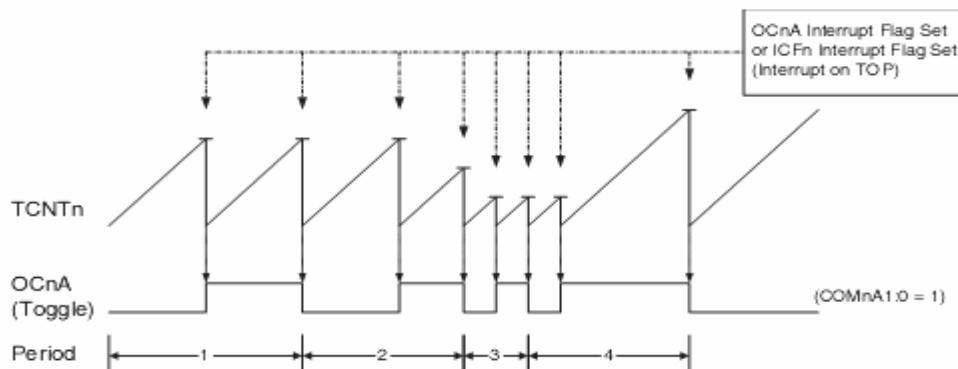
Các bộ phận so sánh đầu ra có thể được sử dụng để phát ra các ngắt tại một vài thời điểm định trước .Việc sử dụng so sánh đầu ra để phát các xung trong chế độ bình thường thì không được khuyến cáo , do đó điều này sẽ làm tiêu tốn nhiều thời gian của CPU

## Chế độ CTC – Clear Timer on Compare Match

Trong chế độ CTC (WGMn3:0=4 hoặc 12), thanh ghi OCRnA hoặc ICRn được sử dụng để điều khiển độ chính xác của bộ đếm. Trong chế độ CTC bộ đếm bị xóa về 0 khi mà giá trị của bộ đếm (TCNTn) tương ứng với một trong hai bit OCRnA (WGMn3:0=4) hoặc ICRn (WGMn3:0=12). Thanh ghi OCRnA hoặc ICRn xác định giá trị TOP cho bộ đếm, và cũng là độ chính xác của nó. Chế độ này cho phép việc điều khiển của tần số đầu ra ghép so sánh tốt hơn. Nó cũng là quá trình điều khiển đơn giản nhất của việc đếm các sự kiện bên ngoài.

Biểu đồ thời gian cho chế độ CTC được chỉ ra trên hình 51. Giá trị của bộ đếm (TCNTn) làm tăng cho đến khi một ghép so sánh xuất hiện với một trong hai thanh ghi OCRnA hoặc ICRn, và sau đó bộ đếm bị xóa

**Figure 51. CTC Mode, Timing Diagram**



Một ngắt có thể được sinh ra tại mỗi lần mà giá trị của bộ đếm đạt đến giá trị TOP bằng việc sử dụng cờ OCFnA hoặc ICFn theo thanh ghi được sử dụng để xác định giá trị TOP. Nếu ngắt được kích hoạt, chương trình con điều khiển ngắt có thể được sử dụng cho việc cập nhật giá trị TOP. Tuy nhiên, việc thay đổi giá trị TOP đến 1 giá trị đóng lên BOTTOM khi mà bộ đếm đang chạy với 1 hoặc 0 mức thấp của bộ đếm gộp trước phải được thực hiện với sự cẩn thận do chế độ CTC không có tính năng bộ đếm kép. Nếu một giá trị mới được ghi lên thanh ghi OCRnA hoặc thanh ghi ICRn thấp hơn giá trị hiện hành của TCNTn, bộ đếm sẽ bị lỗi ghép so sánh. Bộ đếm sau đó sẽ phải đếm từ giá trị cực đại của nó (0xFFFF) và vùng xung quanh giá trị khởi động tại 0x0000 trước khi ghép so sánh có thể xuất hiện. Trong nhiều trường hợp tính năng này không được xét đến. 1 sự đảo ngược sau đó sẽ sử dụng chế độ fast PWM sử dụng OCRnA cho việc xác định giá trị TOP (WGMn3:0=15) do đó OCRnA sau đó sẽ được ghi vào bộ đếm kép

Về việc phát ra 1 đầu ra dạng sóng trong chế độ CTC, đầu ra OcnA có thể được cài đặt để di chuyển mức logic của nó trên mỗi ghép so sánh bằng việc cài đặt các bit chế độ đầu ra so sánh để thay đổi chế độ (COMnA1:0)=1. Giá trị của OcnA sẽ không được nhìn thấy trên chân cổng trừ phi sự định hướng dữ liệu cho chân cổng được cài đặt lên đầu ra (DDR\_OcnA =1). Dạng sóng được phát ra sẽ có tần số lớn nhất của  $f_{OcnA} =$

$f_{clk\_I/O}/2$  khi mà OCRnA được cài đặt là 0 (0x0000) . Tần số dạng sóng được xác định bởi công thức dưới đây :

$$f_{OCnA} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$

Biến N được đưa ra theo tỉ lệ (1 , 8, 64 , 256 , 1024 )

Như trong chế độ điều khiển bình thường , cờ TOVn được cài đặt trong cùng chu kì xung nhịp mà bộ đếm đếm từ giá trị Max về 0x0000

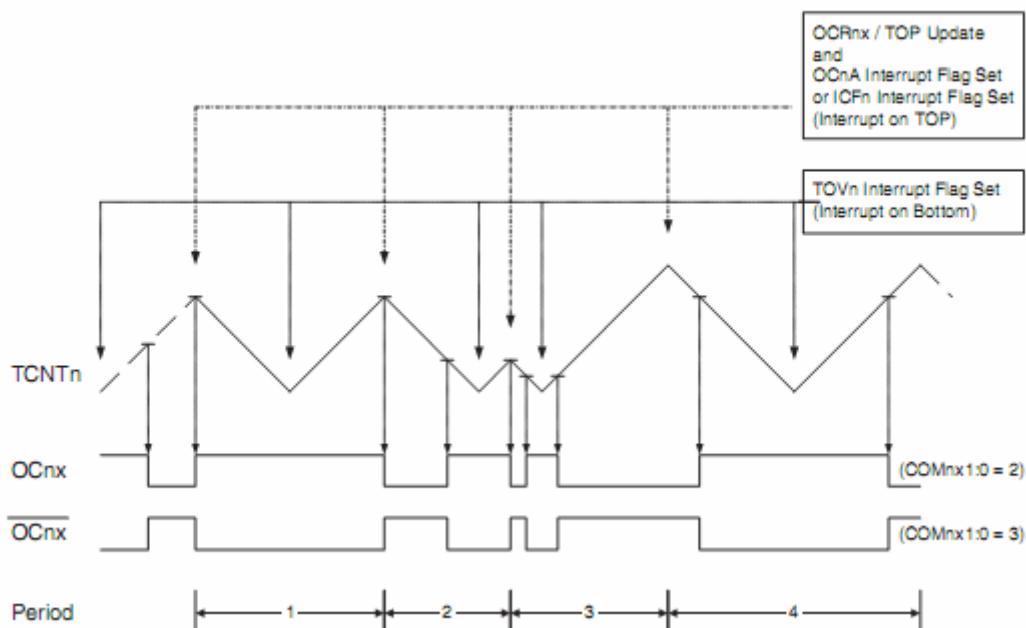
## Chế độ fast PWM

Chế độ điều chế độ rộng xung nhanh hay fast PWM (WGMn3:0 = 5,6,7,14 hoặc 15 ) cung cấp sự một sự phát dạng sóng tần số cao tùy chọn . Chế độ fast PWM khác với các chế độ PWM khác bởi quá trình điều khiển sườn đơn của nó . Bộ đếm đếm từ BOTTOM đến TOP sau đó khởi động lại từ BOTTOM Trong chế độ đầu ra so sánh không đảo , đầu ra so sánh (OCnx) bị xóa trên ghép so sánh giữa TCNTn và OCRnx , và cài đặt ở BOTTOM . Trong chế độ đầu ra so sánh ảo được cài đặt trên ghép so sánh và bị xóa ở BOTTOM . Tương xứng với quá trình điều khiển sườn đơn , tần số hoạt động của chế độ fast PWM có thể cao gấp đôi các chế độ PWM đúng pha đúng tần số và chế độ PWM đúng pha cái mà sử dụng quá trình điều khiển 2 sườn . Tần số cao này làm cho chế độ fast PWM thích hợp tốt với các nguồn thông thường , nguồn chỉnh lưu và các ứng dụng DAC . Tần số cao này cho phép các thành phần vật lí cỡ nhỏ bên trong (cuộn dây , và tụ điện ...) , từ đó giảm giá thành hệ thống .

Độ chính xác của xung PWM cho chế độ fast PWM có thể ổn định ở 8,9 hoặc 10bit , hoặc được xác định bằng thanh ghi ICRn hoặc OCRnA . Độ chính xác cực tiểu cho phép là 2 bit (ICRn hoặc OCRnA đặt là 0x0003) , và độ chính xác cực đại là 16bit (ICRn hoặc OCRnA đặt là MAX) . Độ chính xác của PWM trong các bit có thể được tính toán bằng việc sử dụng công thức dưới đây :

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$

Trong chế độ fast PWM bộ đếm thì được làm tăng cho đến khi giá trị của bộ đếm tương ứng với 1 trong những giá trị ổn định 0x00FF , 0x01FF , hoặc 0x03FF (WGM3:0 =5,6,7 ) , giá trị trong ICRn (WGM3:0 =14) , hoặc giá trị trong OCRnA (WGM3:0 = 15) . Bộ đếm sau đó bị xóa tại chu kì xung nhịp timer tiếp theo . Biểu đồ thời gian cho chế độ fast PWM được chỉ ra trong hình 52 . Hình này chỉ ra chế độ PWM khi mà các bit OCRnA hoặc ICRnA được sử dụng để xác định giá trị TOP . Giá trị của TCNTn trong gián đồ thời gian được chỉ ra như 1 biểu đồ để minh họa cho quá trình điều khiển sườn đơn . Sơ đồ bao gồm các đầu ra đảo và không đảo . Đường nằm ngang đánh dấu trên sườn TCNTn đưa ra trên các ghép so sánh OCRnx và TCNTn . Các cờ báo ngắt OCnx sẽ được cài đặt khi 1 ghép so sánh xuất hiện .

**Figure 53. Phase Correct PWM Mode, Timing Diagram**

Cờ báo tràn Timer/Counter (TOV<sub>n</sub>) được cài đặt mỗi lần bộ đếm đạt đến BOTTOM . Khi một trong hai thanh ghi OCRnA hoặc ICRn được sử dụng cho việc xác định giá trị TOP , cờ ICFn hoặc OCnA được cài đặt tại cùng chu kì xung nhịp như là các thanh ghi được cập nhật với giá trị của bộ đếm kép (TOP) . Các cờ báo ngắn có thể được sử dụng để sinh ra một ngắt mỗi lần bộ đếm đạt đến giá trị TOP hoặc BOTTOM .

Khi thay đổi giá trị TOP chương trình phải đảm bảo rằng giá trị TOP mới cao hơn hoặc bằng giá trị TOP của tất cả các thanh ghi so sánh . Nếu giá trị TOP thấp hơn bất cứ thanh ghi so sánh nào , 1 ghép so sánh sẽ không bao giờ xuất hiện TCNT<sub>n</sub> và OCR<sub>n</sub>x . Chú ý rằng khi sử dụng các giá trị TOP ổn định , các bit không được sử dụng bị che bởi 0 khi bắt cứ thanh ghi OCR<sub>n</sub>x nào được ghi . Như là chu kì thứ 3 được chỉ ra trong hình 53 , việc thay đổi giá trị TOP tích cực trong khi Timer/Counter đang chạy trong chế độ đúng pha có thể gây ra trong một đầu ra không đổi xứng . Kết quả của việc này có thể được tìm thấy trong thời gian của việc cập nhật cho thanh ghi OCR<sub>n</sub>x . Do đó sự cập nhật thanh ghi OCR<sub>n</sub>x tại giá trị TOP , chu kì PWM bắt đầu và kết thúc tại giá trị TOP . Điều này đưa đến chiều dài của sườn xuống được xác định bằng giá trị TOP trước , trong khi độ dài của sườn lên được xác định bằng giá trị TOP mới . Khi cả hai giá trị này khác nhau trên 2 sườn khác nhau của chu kì sẽ khác nhau về độ dài . Sự khác nhau về độ dài đưa ra kết quả không đổi xứng trên đầu ra .

Nó thì được khuyến cáo để sử dụng chế độ đúng pha và đúng tần số để thay thế cho chế độ đúng pha khi thay đổi giá trị TOP trong khi Timer/Counter đang chạy . Khi sử dụng 1 giá trị TOP cố định thực tế không có sự khác nhau giữa hai chế độ của quá trình điều khiển .

Trong chế độ PWM đúng pha , các bộ phận so sánh cho phép việc phát ra các xung PWM trên các chân OC<sub>n</sub>x . Việc cài đặt các bit COMnx1:0 lên 2 sẽ làm giảm 1 xung PWM không đảo và một đầu ra PWM có thể được sinh ra bằng việc cài đặt bit

OCMnx1:0 lên 3 (xem bảng 60 trên trang 134 ). Giá trị thật của OCnx sẽ chỉ được nhìn thấy trên chân cổng nếu sự định hướng dữ liệu cho chân cổng được cài đặt như là đầu ra (DDR\_OCnx) . Dạng xung PWM được sinh ra bằng việc cài đặt (hoặc xóa ) thanh ghi OCnx ở ghép so sánh giữa OCRnx và TCNTn khi bộ đếm tăng , và bị xóa (hoặc cài đặt ) khi thanh ghi OCnx tại ghép so sánh giữa OCRnx và TCNTn khi bộ đếm giảm dần . Tần số xung PWM cho đầu ra khi cài sử dụng chế độ PWM đúng pha có thể được tính bằng công thức dưới đây :

$$f_{OCnxPCPWM} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot TOP}$$

Biến N được đưa ra theo tỉ lệ (1 , 8 , 64 , 256 , 1024)

Giá trị cực biên của thanh ghi OCRnx đưa ra trong trường hợp đặc biệt khi mà sự phát 1 đầu ra xung PWM trong chế độ PWM đúng pha . Nếu thanh ghi OCRnx được cài đặt bằng giá trị BOTTOM , đầu ra sẽ tiếp tục ở mức thấp và nếu cài đặt bằng giá trị TOP thì đầu ra sẽ tiếp tục ở mức cao cho chế độ PWM không đảo . Cho chế độ PWM đảo , đầu ra sẽ có các giá trị logic đối lập .

Nếu OCnA được sử dụng để xác định giá trị TOP (WMGn3:0 = 11) và OCMnA1:0=1 ,đầu ra OCnA sẽ di chuyển với 1 chu kì tải 50%

### Chế độ PWM đúng pha và đúng tần số

Điều chế độ rộng xung đúng pha đúng tần số ,(WGM3:0 =8 hoặc 9 )cung cấp dạng xung PWM lựa chọn đúng pha đúng tần số . Chế độ PWM đúng pha đúng tần số thì giống chế độ PWM đúng pha , đều dựa trên quá trình điều khiển 2 sườn . Bộ đếm đếm lặp lại từ BOTTO (0x0000) đến TOP và sau đó đếm từ TOP về BOTTOM . Trong chế độ đầu ra so sánh không đảo , đầu ra so sánh (OCnx) bị xóa trên ghép so sánh giữa TCNTn và OCRnx trong khi đang đếm lên , và đặt trên ghép so sánh trong khi đang đếm xuống . Trong chế độ đầu ra so sánh đảo , quá trình điều khiển bị đảo ngược . Quá trình điều khiển 2 sườn đưa ra 1 đặc tính đối xứng của các chế độ PWM hai sườn , các chế độ này được ưa thích dùng cho các ứng dụng điều khiển động cơ

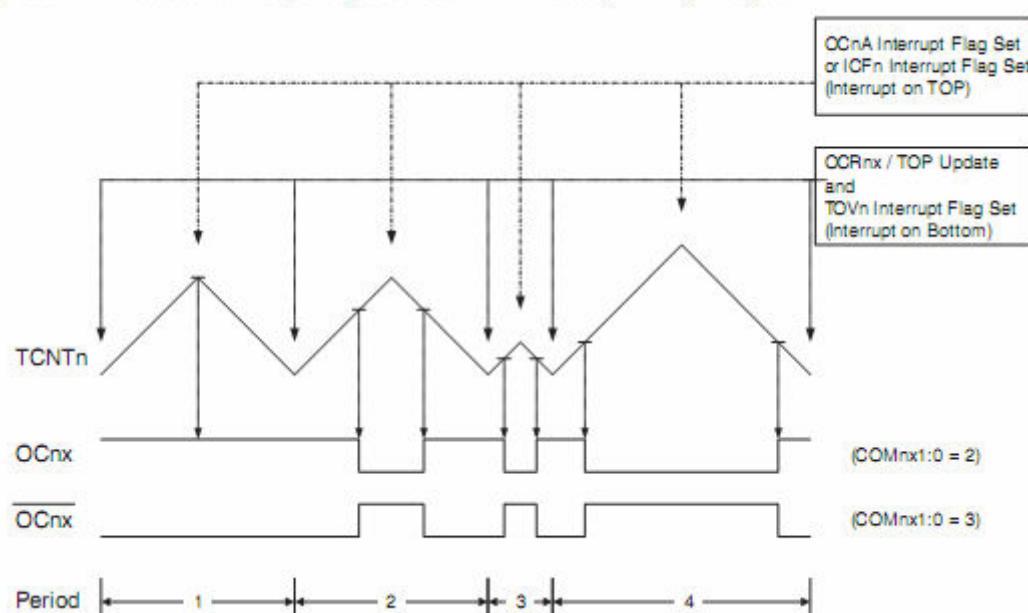
Sự khác nhau chính giữa chế độ đúng pha và chế độ đúng pha, đúng tần số là thời gian mà thanh ghi OCRnx được cập nhật bằng thanh ghi bộ đếm OCRnx (xem hình 53 và hình 54 )

Độ chính xác của chế độ đúng pha , đúng tần số có thể được xác định bằng 1 trong 2 ICRn hoặc OCRnA . Độ chính xác cực tiêu cho phép là 2-bit (ICRn hoặc OCRnA đặt là 0x0003) và độ chính xác lớn nhất là 16bit (ICRn và OCRnA đặt là MAX ) . Độ chính xác của PWM trong các bit có thể được tính toán bằng công thức dưới đây :

$$R_{PFCPWM} = \frac{\log(TOP + 1)}{\log(2)}$$

Trong chế độ đúng pha, đúng tần số, bộ đếm đếm tăng dần cho đến khi giá trị của bộ đếm tương ứng với 1 trong 2 giá trị trong ICRn ( $WGMn3:0 = 8$ ) hoặc giá trị trong OCRnA ( $WGMn3:0 = 9$ ). Bộ đếm sau khi vừa đạt đến giá trị TOP và chuyển hướng đếm. Giá trị của TCNTn sẽ bằng với TOP trong 1 chu kỳ xung nhịp. Biểu đồ thời gian cho chế độ PWM đúng pha và đúng tần số được chỉ ra trong hình 54. Hình vẽ chỉ ra chế độ đúng pha, đúng tần số khi mà OCRnA hoặc ICRn được sử dụng để xác định giá trị TOP. Giá trị của TCNTn trong gián đồ thời gian được chỉ ra như một biểu đồ minh họa cho quá trình điều khiển 2 sườn. Gián đồ bao gồm các đầu ra PWM đảo và không đảo. Đường nằm ngang nhỏ đánh dấu trên sườn TCNTn đưa ra trên ghép so sánh giữa OCnx và TCNTn. Cờ báo ngắn OCnx sẽ được cài đặt khi mà ghép so sánh xuất hiện.

**Figure 54. Phase and Frequency Correct PWM Mode, Timing Diagram**



Cờ báo tràn Timer/Counter (TOVn) được cài đặt tại cùng chu kỳ xung nhịp như là thời gian các thanh ghi được cập nhật với giá trị bộ đếm kép (ở BOTTOM). Khi một trong 2 bit OCRnA và ICRnx được sử dụng cho việc xác định giá trị TOP, cờ OCnA và ICFn cài đặt khi TCNTn đạt đến giá trị TOP. Các cờ báo ngắn sau đó có thể được sử dụng để sinh ra 1 ngắn mỗi lần mà timer đạt đến giá trị TOP và BOTTOM.

Khi thay đổi giá trị TOP chương trình phải đảm bảo rằng giá trị TOP mới cao hơn hoặc bằng giá trị của tất cả các thanh ghi so sánh. Nếu giá trị TOP thấp hơn bất cứ giá trị so sánh thanh ghi nào, 1 ghép so sánh sẽ không bao giờ xuất hiện giữa các TCNTn và OCRnx.

Như hình 54 đã chỉ ra đầu ra được sinh ra, trong sự tương phản với chế độ PWM đúng pha, sự đổi xứng trong tất cả các chu kỳ. Do đó các thanh ghi OCRnx được cập nhật tại BOTTOM, độ dài của sườn lên và xuống sẽ luôn luôn bằng nhau. Điều này đưa các xung đầu ra đổi xứng và vì vậy tần số đúng.

Việc sử dụng các thanh ghi ICRn cho việc xác định giá trị TOP tốt khi sử dụng giá trị ổn định. Bằng việc sử dụng ICRn, thanh ghi OCRnA thì tự do để sử dụng cho việc phát ra 1 xung PWM trên đầu ra OCnA. Tuy nhiên nếu tần số PWM cơ bản được

thay đổi bởi bằng việc thay đổi giá trị TOP , việc sử dụng OCRnA nhu là giá trị TOP rõ ràng là một lựa chọn tốt hơn dù cho đặc tính bộ đệm kép của nó .

Trong chế độ đúng pha và đúng tần số , các bộ phận so sánh cho phép việc phát ra của các xung PWM trên các chân OCnx . Việc cài đặt các bit OCMnx1:0 lên 2 sẽ gây ra 1 1 xung PWM không đảo và đảo có thể được sinh ra bằng việc cài đặt COMnx1:0 lên 3 (xem bảng 60 trang 134). Giá trị thật của OCnx sẽ chỉ được nhìn thấy trên chân cổng nếu sự định hướng dữ liệu cho chân cổng được cài đặt như là cổng ra (DDR\_OCnx ) . Dạng sóng PWM được sinh ra bằng việc cài (hoặc xóa) thanh ghi OCnx tại ghép so sánh giữa OCRnx và TCNTn khi bộ đếm tăng dần , và việc xóa (hoặc cài đặt) thanh ghi OCnx tại ghép so sánh giữa OCRnx và TCNTn khi bộ đếm giảm dần . Tần số PWM cho đầu ra khi sử dụng chế độ đúng pha , đúng tần số có thể được tính bằng công thức dưới đây :

$$f_{OCnxPFCPWM} = \frac{f_{clk\_IO}}{2 \cdot N \cdot TOP}$$

Giá trị cực biên của thanh ghi OCRnx đưa ra trong trường hợp đặc biệt khi mà sự phát 1 đầu ra xung PWM trong chế độ PWM đúng pha . Nếu thanh ghi OCRnx được cài đặt bằng giá trị BOTTOM , đầu ra sẽ tiếp tục ở mức thấp và nếu cài đặt bằng giá trị TOP thì đầu ra sẽ tiếp tục ở mức cao cho chế độ PWM không đảo . Cho chế độ PWM đảo , đầu ra sẽ có các giá trị logic đổi lập .

Nếu OCnA được sử dụng để xác định giá trị TOP (WMGn3:0 = 11) và OCMnA1:0=1 ,đầu ra OCnA sẽ di chuyển với 1 chu kỳ tải

## Giản đồ thời gian của Timer/Counter

Timer/Counter là thiết kế đồng bộ và chu kỳ xung nhịp timer ( $clk_{Tn}0$ )vì vậy được chỉ ra như tín hiệu kích hoạt xung nhịp trong hình dưới đây . Hình vẽ bao gồm thông tin khi các cờ báo ngắt đã kích hoạt , và khi thanh ghi OCRnx được cập nhật với giá trị bộ đếm OCRnx (chỉ trong các chế độ sử dụng bộ đếm kép ) Hình 55 chỉ ra 1 giản đồ thời gian cho việc cài đặt của OCFnx

**Figure 55.** Timer/Counter Timing Diagram, Setting of OCFnx, no Prescaling

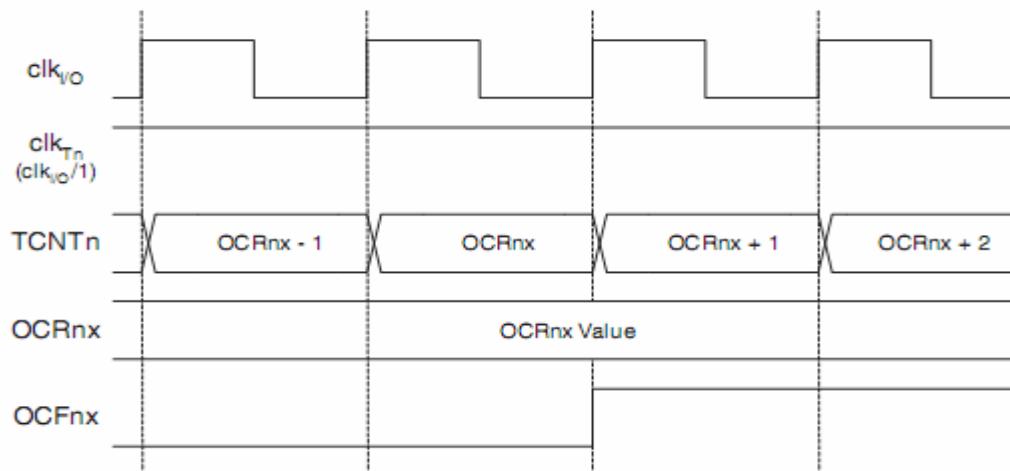
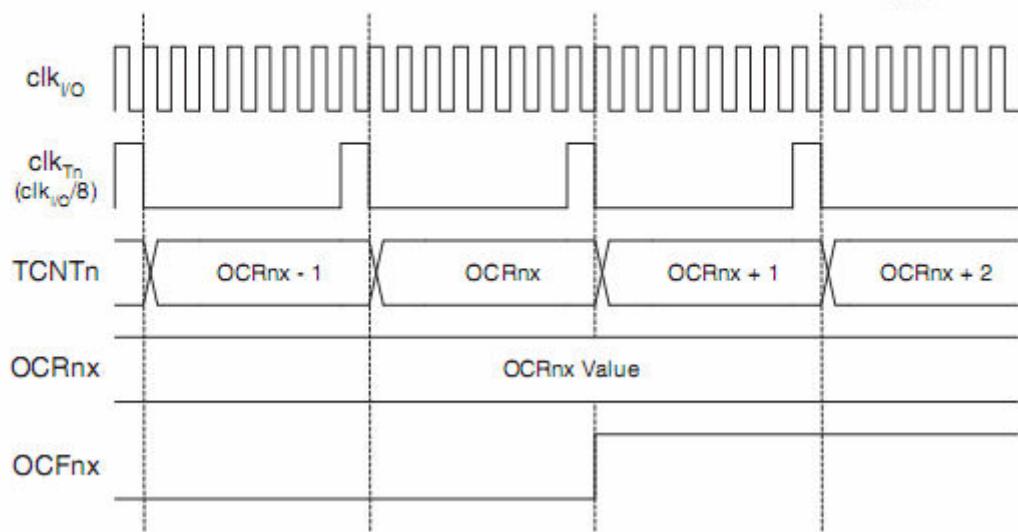


Figure 56 shows the same timing data, but with the prescaler enabled.

**Figure 56. Timer/Counter Timing Diagram, Setting of OCFnx, with Prescaler ( $f_{clk_{I/O}}/8$ )**



Hình 57 chỉ ra chuỗi đếm đóng TOP lên các giá trị thay đổi . Khi sử dụng chế độ PWM đúng pha đúng tần số , các thanh ghi OCRnx được cập nhật tại BOTTOM . giản đồ thời gian sẽ giống nhau , Nhưng giá trị TOP nên được thay đổi bằng BOTTOM , TOP-1 và BOTTOM+1 . Sự giống nhau còn lại áp dụng cho các chế độ mà cài đặt cờ TOVn tại BOTTOM

**Figure 57. Timer/Counter Timing Diagram, no Prescaling**

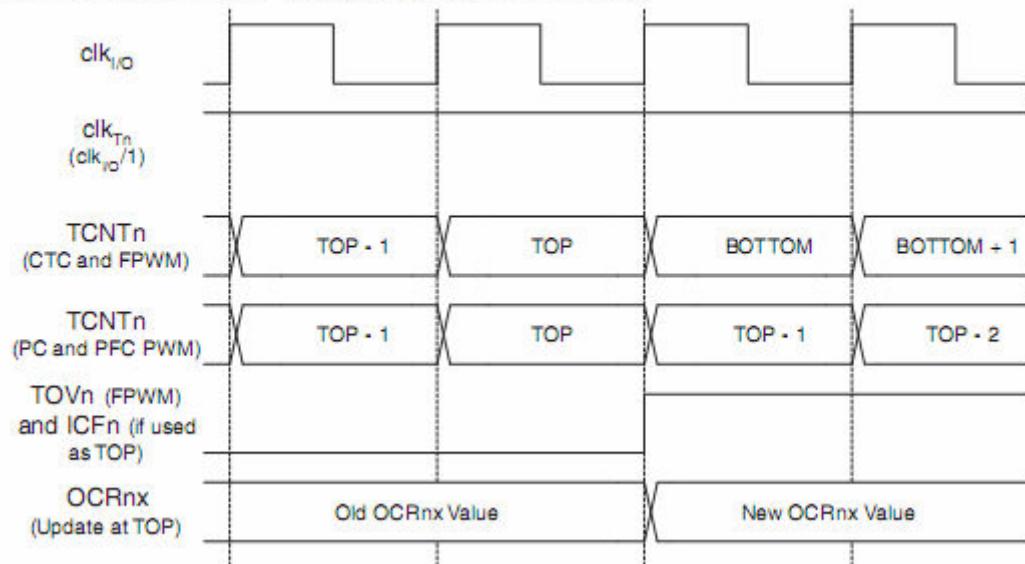
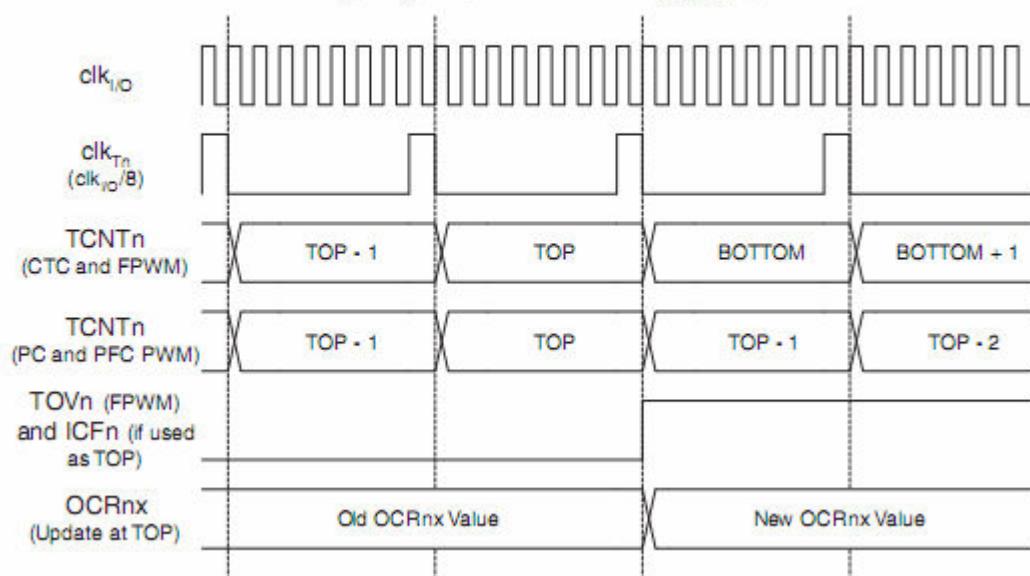


Figure 58 shows the same timing data, but with the prescaler enabled.

**Figure 58. Timer/Counter Timing Diagram, with Prescaler ( $f_{clk_{IO}}/8$ )**



### Sự miêu tả thanh ghi của Timer/Counter 16 bit

Thanh ghi A điều khiển Timer/Counter 1 – TCCR1A

Bit	7	6	5	4	3	2	1	0	TCCR1A
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi A điều khiển Timer/Counter 3 – TCCR3A

Bit	7	6	5	4	3	2	1	0	TCCR3A
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7:6 – COMnA1:0 : chế độ đầu ra so sánh cho kênh A

Bit 5:4 – COMnB1:0 : chế độ đầu ra so sánh cho kênh B

Bit 3:2 – ComnC 1:0 : chế độ đầu ra so sánh cho kênh C

COMnA1:0 , COMnB 1:0 , và COMnC1:0 điều khiển các chân so sánh đầu ra (OcnA , OcnB , và OcnC tương ứng ) . Nếu 1 hoặc cả hai trong số các bit COMnA1:0 được ghi là 1 , đầu ra OcnA ghi đè lên cổng chức năng của chân I/O mà nó được kết nối đến . Nếu 1 hoặc cả hai trong số các bit COMnB1:0 được ghi là 1 , đầu ra OCnB ghi đè lên cổng chức năng của chân I/O mà nó được kết nối đến . Nếu 1 hoặc cả hai trong số các bit COMnC1:0 được ghi là 1 , đầu ra OCnC ghi đè lên cổng chức năng của chân I/O mà nó được kết nối đến . Tuy nhiên , chú ý rằng bit thanh ghi định hướng dữ liệu (DDR) tương ứng với OcnA , OcnB hoặc chân OcnC phải được cài đặt theo thứ tự kích hoạt bộ điều khiển đầu ra

Khi OCnA , OCnB , và OCnC được nối với chân chức năng của các bit COMnx1:0 thì phụ thuộc vào việc cài đặt các bit WGMn3:0 . Bảng 58 chỉ ra bit chức năng COMnx1:0 khi các bit WGMn3:0 được cài đặt tới một chế độ bình thường hoặc chế độ CTC (không phải PWM)

**Table 58.** Compare Output Mode, non-PWM

COMnA1/COMnB1/ COMnC1	COMnA0/COMnB0/ COMnC0	Description
0	0	Normal port operation, OCnA/OCnB/OCnC disconnected.
0	1	Toggle OCnA/OCnB/OCnC on compare match.
1	0	Clear OCnA/OCnB/OCnC on compare match (set output to low level).
1	1	Set OCnA/OCnB/OCnC on compare match (set output to high level).

Bảng 59 chỉ ra các bit chức năng COMnx1:0 khi các bit WGMn3:0 được cài đặt trong chế độ fast PWM .

**Table 59.** Compare Output Mode, Fast PWM

COMnA1/COMnB1/ COMnC1	COMnA0/COMnB0/ COMnC0	Description
0	0	Normal port operation, OCnA/OCnB/OCnC disconnected.
0	1	WGMn3:0 = 15: Toggle OCnA on Compare Match, OCnB/OCnC disconnected (normal port operation). For all other WGMn settings, normal port operation, OCnA/OCnB/OCnC disconnected.
1	0	Clear OCnA/OCnB/OCnC on compare match, set OCnA/OCnB/OCnC at BOTTOM, (non-inverting mode)
1	1	Set OCnA/OCnB/OCnC on compare match, clear OCnA/OCnB/OCnC at BOTTOM, (inverting mode)

Note: A special case occurs when OCRnA/OCRnB/OCRnC equals TOP and COMnA1/COMnB1/COMnC1 is set. In this case the compare match is ignored, but the set or clear is done at BOTTOM. See "Fast PWM Mode" on page 125. for more details.

Bảng 60 chỉ ra các bit chức năng COMnx1:0 khi các bit WGMn3:0 được cài đặt trong chế độ PWM đúng pha và đúng tần số

**Table 60.** Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM

COMnA1/COMnB1/ COMnC1	COMnA0/COMnB0/ COMnC0	Description
0	0	Normal port operation, OCnA/OCnB/OCnC disconnected.
0	1	WGMn3:0 = 9 or 11: Toggle OCnA on Compare Match, OCnB/OCnC disconnected (normal port operation). For all other WGMn settings, normal port operation, OCnA/OCnB/OCnC disconnected.
1	0	Clear OCnA/OCnB/OCnC on compare match when up-counting. Set OCnA/OCnB/OCnC on compare match when downcounting.
1	1	Set OCnA/OCnB/OCnC on compare match when up-counting. Clear OCnA/OCnB/OCnC on compare match when downcounting.

Note: A special case occurs when OCRnA/OCRnB/OCRnC equals TOP and COMnA1/COMnB1//COMnC1 is set. See "Phase Correct PWM Mode" on page 127. for more details.

Bit 1:0 - WGMn1:0 : chế độ phát dạng sóng

Việc kết hợp với các bit WGMn3:2 được tìm thấy trong thanh ghi TCCRnB , các bit này điều khiển các chuỗi đếm của bộ đếm , nguồn cho giá trị bộ đếm cực đại (TOP) , và loại dạng sóng phát ra được sử dụng , xem bảng 61 . Các chế độ điều khiển được hỗ trợ bằng các bộ phận Timer/Counter là : Chế độ normal (counter) , CTC mode (Clear Timer on Compare match) , và 3 loại của chế độ điều chế độ rộng xung PWM ( xem "Modes of Operation " trang 124)

**Table 61.** Waveform Generation Mode Bit Description

Mode	WGMn3	WGMn2 (CTCn)	WGMn1 (PWMrn1)	WGMn0 (PWMrn0)	Timer/Counter Mode of Operation <sup>(1)</sup>	TOP	Update of OCRnx at	TOVn Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCRnA	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICRn	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCRnA	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICRn	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCRnA	TOP	BOTTOM
12	1	1	0	0	CTC	ICRn	Immediate	MAX
13	1	1	0	1	(Reserved)	-	-	-
14	1	1	1	0	Fast PWM	ICRn	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCRnA	BOTTOM	TOP

Note: 1. The CTCn and PWMrn1:0 bit definition names are obsolete. Use the WGMn2:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

## Thanh ghi B điều khiển Timer/Counter 1 – TCCR1B

Bit	7	6	5	4	3	2	1	0	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

## Thanh ghi B điều khiển Timer/Counter 3 – TCCR3B

Bit	7	6	5	4	3	2	1	0	TCCR3B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – ICNCn : khóa cắt nhiễu bộ bắt tín hiệu đầu vào

Việc cài đặt bit này (là 1 ) sẽ kích hoạt khóa cắt nhiễu bộ bắt tín hiệu đầu vào .

Khi khóa cắt nhiễu (Noise Canceler) được kích hoạt , đầu vào từ chân bắt tín hiệu đầu vào (ICPn) được lọc . Chức năng của bộ lọc là cần thiết 4 quá trình lấy mẫu bằng nhau liên tiếp của chân ICPn cho việc thay đổi đầu ra của nó . Bộ bắt tín hiệu đầu vào do vậy bị trễ bởi 4 chu kỳ xung nhịp của bộ tạo dao động khi mà khóa cắt nhiễu được kích hoạt

Bit6 – ICESn : lựa chọn sườn của bộ bắt tín hiệu đầu vào

Bit này lựa chọn sườn nào trên chân cổng của bộ bắt tín hiệu đầu vào (ICPn ) cái mà được sử dụng để khởi động 1 sự kiện truy bắt tín hiệu . Khi bit ICESn được ghi là 0 , 1 sườn xuống (âm ) được sử dụng như là để khởi động , và khi bit ICESn được viết là 1 , 1 sườn lên (đương) sẽ khởi động bộ bắt tín hiệu

Khi 1 bộ bắt tín hiệu được khởi động theo việc cài đặt bit ICESn , giá trị của bộ đếm được sao chép vào trong thanh ghi truy bắt tín hiệu đầu vào (ICRn) . Biến cố cũng sẽ được cài đặt cờ báo truy bắt tín hiệu đầu vào(ICFn), và điều này có thể được sử dụng để gây ra ngắt truy bắt tín hiệu đầu vào , nếu ngắt này được kích hoạt

Khi ICRn được sử dụng như giá trị TOP ( xem sự mô tả của các bit WGMn3:0 đặt trong thanh ghi TCCRnA và TCCRnB ) , ICPn được ngắt kết nối và do đó chức năng truy bắt tín hiệu đầu ra bị vô hiệu hóa .

Bit 5 – bit dự trữ

Bit này được dự trữ cho việc sử dụng trong tương lai . Để đảm bảo tương thích với các thiết bị trong tương lai, bit này phải được viết là 0 khi mà thanh ghi TCCRnB được ghi

Bit 4:3 - WGMn3:2 : chế độ phát dạng sóng

Xem phần mô tả thanh ghi TCCRnA

Bit2:0 – CSn2:0 : lựa chọn xung nhịp

3 bít lựa chọn xung nhịp lựa chọn nguồn xung nhịp được sử dụng bởi Timer/Counter , xem hình 55 và 56

**Table 62. Clock Select Bit Description**

CSn2	CSn1	CSn0	Description
0	0	0	No clock source. (Timer/Counter stopped)
0	0	1	clk <sub>IO</sub> /1 (No prescaling)
0	1	0	clk <sub>IO</sub> /8 (From prescaler)
0	1	1	clk <sub>IO</sub> /64 (From prescaler)
1	0	0	clk <sub>IO</sub> /256 (From prescaler)
1	0	1	clk <sub>IO</sub> /1024 (From prescaler)
1	1	0	External clock source on Tn pin. Clock on falling edge
1	1	1	External clock source on Tn pin. Clock on rising edge

Nếu các chế độ chân bên ngoài được sử dụng cho Timer/Counter , sự chuyển tiếp trên các chân Tn sẽ khóa bộ đếm dù cho chân được cấu hình như là 1 đầu ra . Đặc điểm này cho phép phần mềm điều khiển việc đếm

### Thanh ghi C điều khiển Timer/Counter 1 – TCCR1C

Bit	7	6	5	4	3	2	1	0	TCCR1C
Read/Write	W	W	W	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

### Thanh ghi C điều khiển Timer/Counter 3 – TCCR3C

Bit	7	6	5	4	3	2	1	0	TCCR3C
Read/Write	W	W	W	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – FOCnA : so sánh đầu ra cường bức cho kênh A

Bit 6 – FOCnB : so sánh đầu ra cường bức cho kênh B

Bit 5 – FOCnC : so sánh đầu ra cường bức cho kênh C

Các bit FOCnA , FOCnB , FOCnC chỉ hoạt động khi các bit WGMn3:0 xác định 1 chế độ không phải là PWM . Khi việc viết 1 mức logic 1 lên các bit FOCnA , FOCnB , FOCnC , 1 ghép so sánh trung gian bị cường ép trên bộ phận phát dạng sóng . Đầu ra FOCnA , FOCnB , FOCnC bị thay đổi theo việc cài đặt các bit WGMn1:0 của nó . Chú ý rằng các bit FOCnA , FOCnB , FOCnC được cài đặt như là đầu đo (strobe) . Vì vậy nó là giá trị đưa ra trên các bit OCMnx1:0 cái mà xác định ảnh hưởng của so sánh cường bức .

1 đầu dò FOCnA , FOCnB , FOCnC sẽ không sinh ra bất cứ ngắt nào nó sẽ xóa Timer trong chế độ CTC mode bằng việc sử dụng OCRnA như là giá trị TOP .

Các bit FOCnA , FOCnB , FOCnC thì luôn đọc là 0

Bit 4:0 – các bit dự trữ

Các bit này được dự trữ cho việc sử dụng trong tương lai . Để đảm bảo tương thích với các thiết bị trong tương lai, bit này phải được viết là 0 khi mà thanh ghi TCCRnC được ghi

## Timer/Counter1 – TCNT1H và TCNT1L

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	TCNT1H							
Initial Value	0	0	0	0	0	0	0	0	TCNT1L

## Timer/Counter 3 – TCNT3H và TCNT3L

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	TCNT3H							
Initial Value	0	0	0	0	0	0	0	0	TCNT3L

Hai vùng địa chỉ I/O Timer/Counter (TCNTnH và TCNTnL , được kết hợp TCNTn) đưa ra sự truy nhập trực tiếp , cả hai đều phục vụ cho quá trình đọc và ghi , tới bộ phận Timer/Counter của bộ đếm 16bit . Để đảm bảo rằng các byte cao và thấp được đọc và ghi một cách đồng thời khi CPU truy cập vào các thanh ghi này , sự truy nhập được tiến hành sử dụng 1 thanh ghi byte cao tạm thời 8bit (TEMP) . Thanh ghi tạm thời này được chia sẻ bởi tất cả các thanh ghi 16bit . Xem “Accessing 16bit Register” trên trang 115 .

Sự sửa đổi bộ đếm (TCNTn) trong khi bộ đếm đang chạy sẽ đưa ra 1 nguy cơ của việc lỗi 1 ghép so sánh giữa TCNTn và 1 trong các thanh ghi OCRnx

Việc viết thanh ghi TCNTn khóa (gỡ bỏ ) ghép so sánh trên các xung nhịp timer dưới đây cho tất cả các bộ phận so sánh .

## Thanh ghi so sánh đầu ra 1A – OCR1AH và OCR1AL

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCR1AH							
Initial Value	0	0	0	0	0	0	0	0	OCR1AL

## Thanh ghi so sánh đầu ra 1B – OCR1BH và OCR1BL

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCR1BH							
Initial Value	0	0	0	0	0	0	0	0	OCR1BL

## Thanh ghi so sánh đầu ra 1C – OCR1CH và OCR1CL

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCR1CH							
Initial Value	0	0	0	0	0	0	0	0	OCR1CL

### Thanh ghi so sánh đầu ra 3 A – OCR3AH và OCR3AL

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCR3AH							
Initial Value	0	0	0	0	0	0	0	0	OCR3AL

### Thanh ghi so sánh đầu ra 3B – OCR3BH và OCR3BL

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCR3BH							
Initial Value	0	0	0	0	0	0	0	0	OCR3BL

### Thanh ghi so sánh đầu ra 3C – OCR3CH và OCR3CL

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCR3CH							
Initial Value	0	0	0	0	0	0	0	0	OCR3CL

Các thanh ghi so sánh đầu ra bao gồm 1 giá trị 16bit mà được so sánh liên tiếp với giá trị đếm (TCNTn) . Một ghép có thể được sử dụng để phát ra 1 ngắt so sánh đầu ra , hoặc phát ra 1 đầu ra dạng sóng trên chân Ocnx .

Các thanh ghi so sánh đầu ra là cỡ 16 bit . Để đảm bảo rằng các byte cao và thấp được đọc và ghi một cách đồng thời khi CPU truy cập vào các thanh ghi này , sự truy nhập được tiến hành sử dụng 1 thanh ghi byte cao tạm thời 8bit (TEMP) . Thanh ghi tạm thời này được chia sẻ bởi tất cả các thanh ghi 16bit . Xem “Accessing 16bit Register” trên trang 115 .

### Thanh ghi1 truy bắt tín hiệu đầu vào – ICR1H và ICR1L

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	ICR1H							
Initial Value	0	0	0	0	0	0	0	0	ICR1L

### Thanh ghi truy bắt tín hiệu đầu vào 3 – ICR3H và ICR3L

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	ICR3H							
Initial Value	0	0	0	0	0	0	0	0	ICR3L

Bộ truy bắt tín hiệu đầu vào được cập nhật với giá trị bộ đếm (TCNTn) mỗi lần mà biến cố xuất hiện trên chân ICPn ( hoặc lựa chọn trên đầu ra bộ so sánh tương tự cho Timer/Counter1) . Bộ truy bắt tín hiệu đầu vào có thể được sử dụng để xác định giá trị TOP của bộ đếm .

Thanh ghi truy bắt tín hiệu đầu vào là cỡ 16bit . Để đảm bảo rằng các byte cao và thấp được đọc và ghi một cách đồng thời khi CPU truy cập vào các thanh ghi này , sự truy nhập được tiến hành sử dụng 1 thanh ghi byte cao tạm thời 8bit (TEMP) . Thanh ghi tạm thời này được chia sẻ bởi tất cả các thanh ghi 16bit . Xem “Accessing 16bit Register” trên trang 115 .

## Thanh ghi che ngắt Timer/Counter – TIMSK

Bit	7	6	5	4	3	2	1	0	TIMSK
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Note: This register contains interrupt control bits for several Timer/Counters, but only Timer1 bits are described in this section. The remaining bits are described in their respective timer sections.

Bit 5 – TICIE1 : Timer/Counter 1 , kích hoạt ngắt bộ truy bắt tín hiệu vào

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt kích hoạt chung ) , Timer/Counter 1 ngắt bộ truy bắt tín hiệu đầu vào được kích hoạt . Các vecto ngắt tương ứng (xem Interrupt trang 60 ) được thực thi khi cờ báo ICF1 ,đặt trong thanh ghi TIFR được cài đặt

Bit 4 – OCIE1A : Timer/Counter 1 , kích hoạt ngắt ghép so sánh đầu ra A

Khi bit này được ghi là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt ) , Ngắt ghép so sánh đầu ra A Timer/Counter1 được kích hoạt . Các vec tơ ngắt tương ứng ( xem trang 60 ) được thực thi khi cờ báo OCF1A , đặt trong thanh ghi TIFR được cài đặt .

Bit 3 – OCIE1B : kích hoạt ghép so sánh đầu ra B , Timer/Counter1

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt ) 1 ngắt ghép so sánh đầu ra B Timer/Counter1 được kích hoạt . Các vecto ngắt tương ứng được thực thi khi mà cờ báo OCF1A , đặt trong TIFR được cài đặt

Bit 2 – TOIE1 : kích hoạt ngắt tràn , Timer/Counter1

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt) , ngắt tràn Timer/Counter1 được kích hoạt . Các vecto ngắt tương ứng (xem trang 60 ) được thực thi khi cờ báo TOV1 , đặt trong thanh ghi TIFR được cài đặt .

## Thanh ghi che ngắt Timer/Counter mở rộng – ETIMSK

Bit	7	6	5	4	3	2	1	0	
Read/Write	-	-	TICIE3	OCIE3A	OCIE3B	TOIE3	OCIE3C	OCIE1C	ETIMSK
Initial Value	0	0	0	0	0	0	0	0	

Note: This register is not available in ATmega103 compatibility mode.

Bit 7:6 – các bit dự trữ

Các bit này được dự trữ cho việc sử dụng trong tương lai . Để đảm bảo tương thích với các thiết bị trong tương lai, bit này phải được viết là 0 khi mà thanh ghi ETIMSK được ghi

Bit 5 – TICIE3 : kích hoạt ngắt bộ truy bắt tín hiệu đầu vào

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt) , ngắt truy bắt tín hiệu đầu ra Timer/Counter3 được kích hoạt . Các vecto ngắt tương ứng (xem trang 60 ) được thực thi khi cờ báo ICF3, đặt trong thanh ghi ETIFR được cài đặt .

Bit4 – OCIE3A : Timer/Counter3 , kích hoạt ngắt ghép đầu ra so sánh A

Khi bit này được ghi là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt ) , Ngắt ghép so sánh đầu ra A Timer/Counter3 được kích hoạt . Các vec tơ ngắt tương ứng ( xem trang 60 ) được thực thi khi cờ báo OCF3A , đặt trong thanh ghi ETIFR được cài đặt .

Bit 3 – OCIE3B : Timer/Counter3 , kích hoạt ngắt ghép đầu ra so sánh B

Khi bit này được ghi là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt ) , Ngắt ghép so sánh đầu ra B Timer/Counter3 được kích hoạt . Các vec tơ ngắt tương ứng ( xem trang 60 ) được thực thi khi cờ báo OCF3B , đặt trong thanh ghi ETIFR được cài đặt

Bit 2 – TOIE3 , Timer /Counter3 , kích hoạt ngắt tràn

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt) , ngắt tràn Timer/Counter3 được kích hoạt . Các vecto ngắt tương ứng (xem trang 60 ) được thực thi khi cờ báo TOV3 , đặt trong thanh ghi ETIFR được cài đặt .

Bit 1 – OCIE3C : Timer /Counter3 , kích hoạt ngắt ghép so sánh đầu ra C

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt) , ngắt ghép so sánh đầu ra C của Timer/Counter3 được kích hoạt . Các vecto ngắt tương ứng (xem trang 60 ) được thực thi khi cờ báo OCF3C , đặt trong thanh ghi ETIFR được cài đặt .

Bit 0 – OCIE1C : Timer/Counter1 , kích hoạt ngắt ghép so sánh đầu ra C

Khi bit này được viết là 1 , và cờ I trong thanh ghi trạng thái được cài đặt ( các ngắt chung đã kích hoạt) , ngắt ghép so sánh đầu ra C của Timer/Counter1 được kích hoạt . Các vecto ngắt tương ứng (xem trang 60 ) được thực thi khi cờ báo OCF1C , đặt trong thanh ghi ETIFR được cài đặt

## Thanh ghi cờ báo ngắt Timer/Counter – TIFR

Bit	7	6	5	4	3	2	1	0	TIFR
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Note: This register contains flag bits for several Timer/Counters, but only timer 1 bits are described in this section. The remaining bits are described in their respective timer sections.

Bit 5 – ICF1 : Timer/Counter 1 , cờ báo truy bắt tín hiệu đầu ra

Cờ này được cài đặt khi 1 bộ truy bắt biến có xuất hiện trên chân ICP1 . Khi mà thanh ghi truy bắt tín hiệu đầu vào (ICR1) được cài đặt bởi bit WGMn3:0 được sử dụng như là giá trị TOP , cờ ICF1 được cài đặt khi bộ đếm tiến tới giá trị TOP

ICF1 bị xóa 1 cách tự động khi vecto ngắt truy bắt đầu vào được thực thi . Như một sự lựa chọn , ICF1 có thể bị xóa bằng việc viết 1 mức logic 1 lên vùng nhớ bit của nó .

Bit 4 – OCF1A : Timer/Counter1 , cờ ghép so sánh đầu ra A

Cờ này được cài đặt trong chu kỳ xung nhịp timer sau khi giá trị của bộ đếm tương ứng với thanh ghi so sánh đầu ra A (OCR1A )

Chú ý rằng 1 đầu dò so sánh đầu ra cường bức (FOC1A) sẽ không cài đặt cờ OCF1A

OCF1A bị xóa 1 cách tự động khi mà vecto ngắt ghép đầu ra so sánh A được thực thi . Như một sự lựa chọn, OCF1A có thể bị xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó .

Bit 3 – OCF1B : Timer/Counter1 , cờ báo ghép so sánh đầu ra B

Cờ này được cài đặt trong chu kỳ xung nhịp timer sau khi giá trị bộ đếm (TCNT1) tương ứng với thanh ghi so sánh đầu ra B (OCR1B)

Chú ý rằng đầu dò so sánh đầu ra cường bức (FOC1B ) sẽ không cài đặt cờ OCF1B

OCF1B bị xóa 1 cách tự động khi mà vecto ngắt ghép đầu ra so sánh B được thực thi . Như một sự lựa chọn, OCF1B có thể bị xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó .

Bit 2 – TOV1 : Timer/Counter1 , cờ tràn

Việc cài đặt cờ này thì phụ thuộc vào việc cài đặt của các bit WGMn3:0 . Trong chế độ thông thường và chế độ CTC , cờ TOV1 được cài đặt khi timer bị tràn . Tham khảo thêm bảng 61 trang 135 về xử lý cờ tràn TOV1 khi việc sử dụng các cài đặt bit WGMn3:0 khác .

TOV1 thì tự động bị xóa khi mà vec tơ ngắt báo tràn Timer/Counter1 được thực thi . Như một sự lựa chọn , TOV1 có thể được xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó

## Thanh ghi cờ ngắt Timer/Counter mở rộng – ETIFR

Bit	7	6	5	4	3	2	1	0	
Read/Write	-	-	ICF3	OCF3A	OCF3B	TOV3	OCF3C	OCF1C	ETIFR
Initial Value	0	0	0	0	0	0	0	0	

Bit 7:6 – các bit dự trữ

Các bit này được dự trữ cho việc sử dụng trong tương lai . Để đảm bảo tương thích với các thiết bị trong tương lai, bit này phải được viết là 0 khi mà thanh ghi ETIFR được ghi

Bit 5 – ICF3 : Timer/Counter3 , cờ báo truy bắt tín hiệu đầu vào

Cờ này được cài đặt khi 1 bộ truy bắt biến có xuất hiện trên chân ICP3 . Khi mà thanh ghi truy bắt tín hiệu đầu vào (ICR3) được cài đặt bởi bit WGMn3:0 được sử dụng như là giá trị TOP , cờ ICF3 được cài đặt khi bộ đếm tiến tới giá trị TOP

ICF3 bị xóa 1 cách tự động khi vecto ngắt truy bắt đầu vào3 được thực thi . Như một sự lựa chọn , ICF3 có thể bị xóa bằng việc viết 1 mức logic 1 lên vùng nhớ bit của nó

Bit 4 – OCF3A : Timer/Counter 3 , cờ ghép so sánh đầu ra A

Cờ này được cài đặt trong chu kỳ xung nhịp timer sau khi giá trị của bộ đếm tương ứng với thanh ghi so sánh đầu ra A (OCR3A )

Chú ý rằng 1 đầu dò so sánh đầu ra cưỡng bức (FOC3A) sẽ không cài đặt cờ OCF3A

OCF1A bị xóa 1 cách tự động khi mà vecto ngắt ghép đầu ra so sánh3 A được thực thi . Như một sự lựa chọn, OCF3A có thể bị xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó .

Bit 3 – OCF3B : Timer/Counter3 , cờ báo ghép so sánh đầu ra B

Cờ này được cài đặt trong chu kỳ xung nhịp timer sau khi giá trị bộ đếm (TCNT3) tương ứng với thanh ghi so sánh đầu ra B (OCR3B)

Chú ý rằng đầu dò so sánh đầu ra cưỡng bức (FOC3B ) sẽ không cài đặt cờ OCF3B

OCF3B bị xóa 1 cách tự động khi mà vecto ngắt ghép đầu ra so sánh3 B được thực thi . Như một sự lựa chọn, OCF3B có thể bị xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó .

Bit 2 – TOV3 : Timer/Counter3 , cờ tràn

Việc cài đặt cờ này thì phụ thuộc vào việc cài đặt của các bit WGMn3:0 . Trong chế độ thông thường và chế độ CTC , cờ TOV3 được cài đặt khi timer bị tràn . Tham khảo thêm bảng 52 trang 105 về xử lý cờ tràn TOV3 khi việc sử dụng các cài đặt bit WGMn3:0 khác .

TOV3 thì tự động bị xóa khi mà vec tơ ngắt báo tràn Timer/Counter3 được thực thi . Như một sự lựa chọn , TOV3 có thể được xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó

Bit 1 – OCF3C :Timer/Counter3 , cờ ghép so sánh đầu ra C

Cờ này được cài đặt trong 1 chu kì xung nhịp timer sau khi giá trị của bộ đếm (TCNT3) tương ứng với thanh ghi so sánh đầu ra C(OCR3C)

Chú ý rằng 1 đầu dò so sánh đầu ra cưỡng bức (FOC3C) sẽ không cài đặt OCF3C

OCF3C thì tự động được xóa khi vecto ngắn ghép so sánh đầu ra 3 C được thực thi . Như một sự lựa chọn, OCF3C có thể bị xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó

Bit 0 – OCF1C : Timer/Counter1 , cờ ghép so sánh đầu ra C

Cờ này được cài đặt trong một chu kì xung nhịp sau khi giá trị bộ đếm (TCNT1) tương ứng với thanh ghi so sánh đầu ra C (OCR1C)

Chú ý rằng một đầu dò so sánh đầu ra cưỡng bức sẽ không cài đặt cờ OCF1C

OCF1C sẽ tự động bị xóa khi mà vecto ngắn ghép so sánh đầu ra 1 C được thực thi . Như một sự lựa chọn , OCF1C có thể bị xóa bằng việc viết một mức logic 1 lên vùng nhớ bit của nó .

### XIII . Các bộ đếm gộp trước của Timer/Counter 3 ,Timer/Counter 2 và Timer/Counter 1

Các Timer/Counter 3 ,Timer/Counter 2 và Timer/Counter 1 chia sẻ module bộ đếm gộp trước giống nhau , nhưng các Timer/Counter có sự cài đặt bộ đếm gộp trước khác nhau . Sự miêu tả phía dưới được áp dụng cho tất cả các Timer/Counter được nói đến .

#### Nguồn xung nhịp bên trong

Timer/Counter có thể bị khóa trực tiếp bằng xung nhịp hệ thống( bằng việc cài đặt CSn2:0 =1) . Điều này cung cấp sự điều khiển nhanh nhất , với 1 tần số xung nhịp Timer/Counter cực đại bằng với tần số xung nhịp hệ thống( $f_{CLK\_I/O}$ ) . Như một sự lựa chọn, 1 trong 4 đầu ra từ bộ đếm gộp trước có thể được sử dụng như là một nguồn phát xung nhịp . Bộ đếm gộp . Xung nhịp của bộ đếm gộp trước có tần số là 1 trong số  $f_{CLK\_I/O}/8$  ,  $f_{CLK\_I/O}/64$  ,  $f_{CLK\_I/O} / 256$  ,  $f_{CLK\_I/O} /1024$

#### Reset bộ đếm gộp trước

Bộ đếm gộp trước thì đang chạy tự do , ví dụ như quá trình hoạt động độc lập của biến logic lựa chọn xung nhịp của Timer/Counter , và bị chia sẻ bởi Timer/Counter 3 ,Timer/Counter 2 và Timer/Counter 1 . Từ khi bộ đếm gộp trước thì không bị hư hỏng bởi việc lựa chọn xung nhịp của Timer/Counter , trạng thái của bộ đếm gộp trước sẽ phụ thuộc vào các tình huống nơi mà xung nhịp của bộ đếm gộp trước được sử dụng . Một ví dụ của việc đếm gộp trước giả tạo xuất hiện khi mà timer được kích hoạt và được khóa bởi bộ đếm gộp trước( 6>CSn2:0>1 ) . Hệ số của các chu kì xung nhịp hệ thống từ khi timer được kích hoạt đến kết quả đếm đầu tiên xuất hiện có thể là từ 1 đến N + 1 chu kì xung nhịp hệ thống, khi N bằng bộ chia tỉ lệ ( 8, 64,256 ,1024 )

Nếu có thể sử dụng reset bộ đếm gộp trước cho việc đồng bộ hóa Timer/Counter đến quá trình thực thi các chương trình . Tuy nhiên , sự cẩn thận phải được tiến hành nếu Timer/Counter khác cái mà chia sẻ cùng 1 bộ đếm gộp trước cũng sử dụng tỉ lệ . 1 Reset bộ đếm gộp trước sẽ ảnh hưởng đến chu kì đếm gộp trước cho tất cả các Timer/Counter được kết nối với nó .

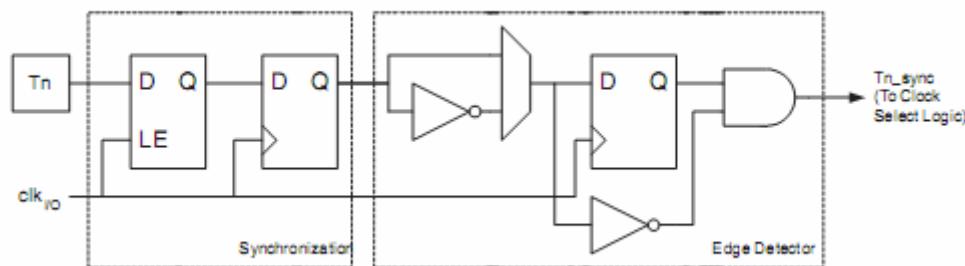
#### Nguồn xung nhịp bên ngoài

Một nguồn xung nhịp bên ngoài áp đặt lên các chân Tn có thể được sử dụng như là xung nhịp Timer/Counter ( $clk_{T_1}$  ,  $clk_{T_2}$  ,  $clk_{T_3}$  ) . Chân Tn thì được lấy mẫu 1 lần trên mỗi chu kì xung nhịp hệ thống bằng chân đồng bộ hóa logic . Tín hiệu đồng bộ hóa (được lấy mẫu ) thì sau đó vượt qua sườn của bộ dò xung . Hình 59 chỉ ra một sơ đồ khói 1chức năng tương đương của bộ đồng bộ hóa Tn và máy dò sườn xung. Các thanh

ghi bị khóa ở sườn dương của xung nhịp hệ thống bên trong ( $\text{clk}_{\text{I/O}}$ ) . Then cài thì được chuyển vào trong chu kì cao của xung nhịp hệ thống bên trong .

Bộ dò sườn phát ra 1 xung  $\text{clk}_{\text{T1}}$ ,  $\text{clk}_{\text{T2}}$ ,  $\text{clk}_{\text{T3}}$  cho mỗi sườn dương ( $\text{CSn2:0} = 7$ ) và âm ( $\text{CSn2:0} = 6$ ) của nó được dò thấy

**Figure 59.** Tn Pin Sampling



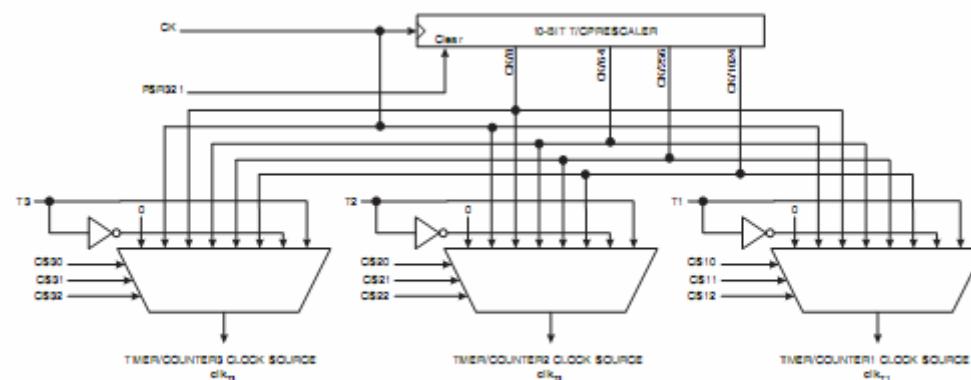
Bộ đồng bộ hóa và các máy dò sườn logic đưa ra 1 độ trễ của 2.5 đến 3.5 lần chu kì xung nhịp hệ thống từ một sườn có thể áp đặt lên chân Tn của bộ đếm được cập nhật .

Việc kích hoạt và vô hiệu hóa của xung nhịp bên ngoài phải được thực hiện khi mà Tn có trạng thái ổn định trong 1 khoảng thời gian nhỏ hơn 1 chu kì xung nhịp hệ thống , nói cách khác nó là 1 nguy cơ cái mà 1 xung nhịp Timer/Counter hỏng được sinh ra .

Mỗi một nửa chu kì xung nhịp bên ngoài được áp dụng phải dài hơn 1 chu kì xung nhịp hệ thống để đảm bảo việc lấy mẫu chính xác . Xung nhịp bên ngoài phải đảm bảo để nhỏ hơn tần số nửa chu kì xung nhịp hệ thống ( $f_{\text{ExtClk}} < f_{\text{clk\_I/O}}/2$ ) đưa ra 50/50 chu kì tải . Vì rằng bộ dò sườn xung sử dụng việc lấy mẫu , tần số cực đại của một xung bên ngoài của nó có thể được dò thấy là 1 nửa tần số lấy mẫu (nguyên lý Nyquist ) . Tuy nhiên , sự thay đổi của tần số xung nhịp hệ thống và chu kì tải gây ra bởi nguồn bộ tạo dao động( thạch anh , bộ cộng hưởng , và tụ điện ) , nó được khuyến cáo rằng tần số lớn nhất của 1 nguồn phát xung nhịp bên ngoài thì nhỏ hơn  $f_{\text{clk\_I/O}}/2.5$

Một nguồn phát xung nhịp ngoài không thể được đếm gộp trước

**Figure 60.** Prescaler for Timer/Counter1, Timer/Counter2, and Timer/Counter3



Note: The synchronization logic on the input pins (T3/T2/T1) is shown in Figure 59.

## Thanh ghi IO chức năng đặc biệt

Bit	7	6	5	4	3	2	1	0	SFIOR
Read/Write	TSM	-	-	-	ACME	PUD	PSR0	PSR321	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – TSM : chế độ đồng bộ hóa Timer/Counter

Việc viết bit TSM là 1 kích hoạt chế độ đồng bộ hóa Timer/Counter . Trong chế độ này, giá trị mà được viết lên các bit PSR0 và PSR321 thì được giữ , do đó việc giữ các tín hiệu reset bộ đếm tương ứng được gán . Điều này đảm bảo rằng các Timer/Counter tương ứng được dừng và có thể được cấu hình lên các giá trị giống nhau mà không có nguy cơ của 1 trong số chúng trong suốt quá trình cấu hình . Khi mà bit TSM được viết là 0 , các bit PSR0 và PSR321 bị xóa bằng phần cứng , và các Timer/Counter bắt đầu việc đếm 1 cách đồng thời .

Bit 0 – PSR321 : bộ đếm gộp trước Reset Timer/Counter 3 ,Timer/Counter 2 và Timer/Counter 1

Khi mà bit này là 1 , bộ đếm gộp trước Timer/Counter 3 ,Timer/Counter 2 và Timer/Counter 1 sẽ được reset . Bit này thông thường được xóa ngay lập tức bằng phần cứng , ngoại trừ nếu bit TSM được cài đặt . Chú ý rằng Timer/Counter 3 ,Timer/Counter 2 và Timer/Counter 1 chia sẻ bộ đếm gộp trước giống nhau và 1 Reset của bộ đếm gộp trước này sẽ hư hỏng trong 3 Timer

## XIV . Timer/Counter 2 8bit với chế độ PWM

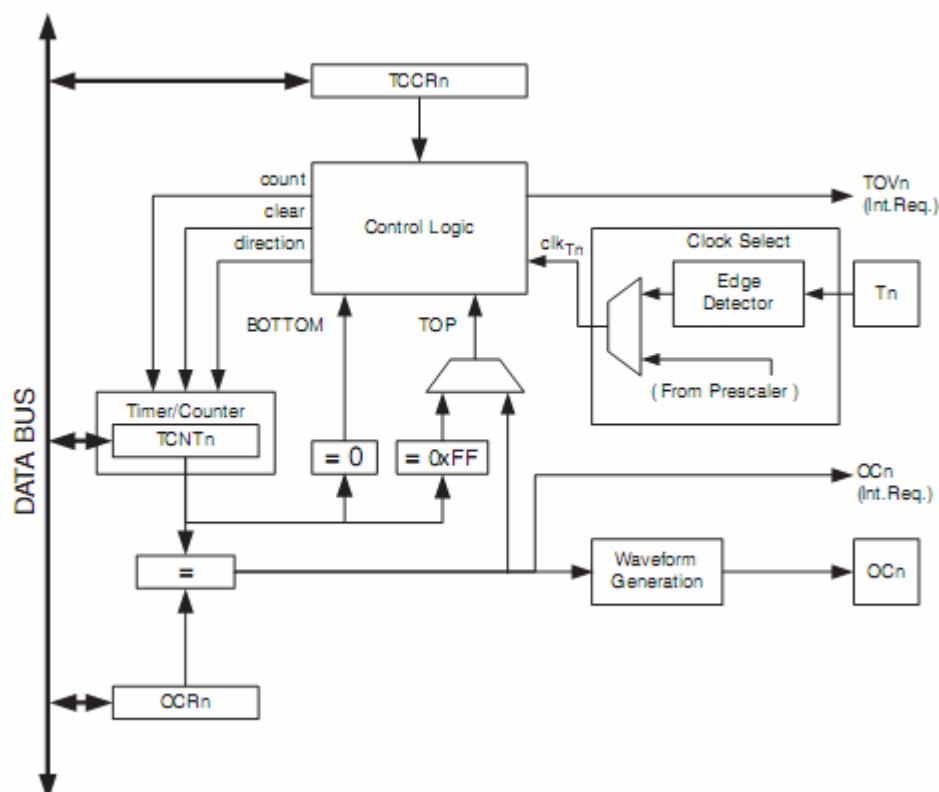
Timer/Counter 2 là 1 module Timer/Counter 2 đa năng dùng chung , kênh đơn . Các đặc điểm chính là :

- Bộ đếm kênh đơn
- Clear Timer on Compare Match (Auto reload )
- Không nhiễu sọc ngang , điều chế độ rộng xung đúng pha ( PWM)
- Máy phát tần số
- Bộ đếm biến cố bên ngoài
- Bộ đếm gộp trước xung nhịp 10bit
- Các nguồn ngắt ghép so sánh và ngắt tràn (TOV2 và OCF2 )

### Tổng quan

Một sơ đồ khôi được rút gọn của Timer/Counter 8bit được chỉ ra trong hình 61 . Về sự cài đặt của các chân I/O , tham khảo “pin Configurations “ trên trang 2 .CPU có thể truy cập vào các thanh ghi I/O bao gồm các bit I/O và các chân I/O , được in đậm . Thanh ghi xác định thiết bị và các vùng bit được liệt kê trong phần mô tả Timer/Counter 8bit trang 157 .

**Figure 61. 8-Bit Timer/Counter Block Diagram**



### Các thanh ghi

Timer/Counter (TCNT2) và thanh ghi so sánh đầu ra (OCR2) là các thanh ghi 8bit .tín hiệu Yêu cầu ngắt (viết tắt là Int.Req trong hình vẽ ) thì được nhìn thấy trong

thanh ghi cờ báo ngắt Timer (TIFR) . Tất cả các ngắt được che riêng biệt với thanh ghi che ngắt Timer (TIMSK) . TIFR và TIMSK thì không được chỉ ra trong hình vẽ do đó các thanh ghi này được chia sẻ bởi các bộ phận timer khác .

Timer/Counter có thể bị khóa bên trong , thông qua bộ đếm gộp trước , hoặc bằng một nguồn xung nhịp bên ngoài trên chân T2 . Khối logic lựa chọn xung nhịp điều khiển nguồn xung nhịp và sườn Timer/Counter sử dụng để tăng (hoặc giảm ) giá trị của nó . Timer/Counter thì không được kích hoạt khi mà không có nguồn xung nhịp nào được lựa chọn . Đầu ra từ khối logic lựa chọn xung nhịp thì được tham khảo như là một xung nhịp timer ( $\text{clk}_{\text{T2}}$  )

Thanh ghi so sánh đầu ra được ghi vào bộ đếm kép (OCR2) thì được so sánh với giá trị của Timer/Counter tại tất các thời điểm . Kết quả của việc so sánh có thể được sử dụng bởi máy phát dạng sóng để phát ra 1 PWM hoặc 1 đầu ra tần số thay đổi trên chân so sánh đầu ra (OC2) . Xem “Output Compare Unit” trên trang 148 để biết thêm chi tiết . Biến cố ghép so sánh cũng sẽ cài đặt cờ so sánh (OCF2) cái mà có thể được sử dụng để phát ra 1 yêu cầu ngắt so sánh đầu ra .

## Các định nghĩa

Nhiều thanh ghi và các bit tham chiếu trong tài liệu này được viết theo mẫu chung . Một trường hợp thấp “n” thay thế cho số thứ tự của Timer/Counter , trong trường hợp 2 . Tuy nhiên khi sử dụng thanh ghi hoặc bit xác định trong 1 chương trình , mẫu chính xác phải được sử dụng (ví dụ như TCNT2 cho việc truy nhập vào giá trị bộ đếm Timer/Counter 2 )

Các định nghĩa trong bảng 63 thì cũng được sử dụng 1 cách rộng rãi xuyên suốt tài liệu này

**Table 63. Definitions**

<b>BOTTOM</b>	The counter reaches the BOTTOM when it becomes 0x00.
<b>MAX</b>	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
<b>TOP</b>	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR2 Register. The assignment is dependent on the mode of operation.

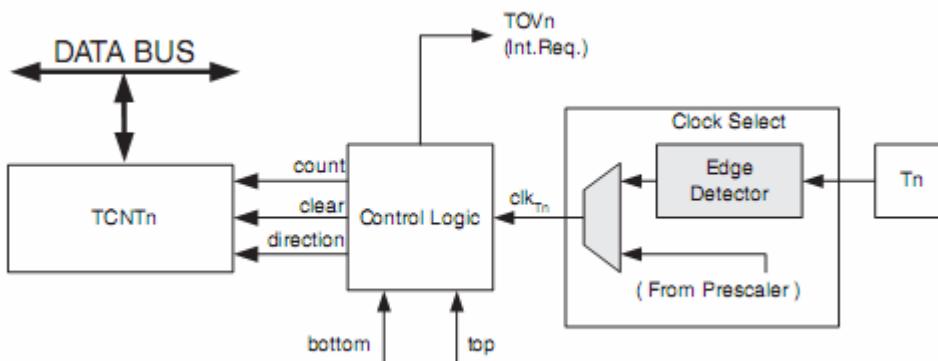
## Các nguồn phát xung nhịp Timer/Counter

Timer/Counter có thể bị khóa bên trong bởi 1 nguồn xung nhịp trong hoặc ngoài . Nguồn xung nhịp được lựa chọn bằng khối logic lựa chọn xung nhịp cái mà được điều khiển bởi các bit lựa chọn xung nhịp(CS22:0) đặt trong thanh ghi điều khiển Timer/Counter (TCCR2) . Để thêm chi tiết về nguồn xung nhịp và bộ đếm gộp trước , xem thêm phần các bộ đếm gộp trước trang 144

## Thành phần bộ đếm

Thành phần chính của bộ đếm 8bit là các bộ phận đếm 2 hướng lập trình được. Hình 62 chỉ ra sơ đồ khối của 1 bộ đếm và các thứ xung quanh nó

**Figure 62. Counter Unit Block Diagram**



Miêu tả tín hiệu (các tín hiệu bên trong )

Count	tăng hoặc giảm TCNT2 bằng 1
Direction	lựa chọn giữa tăng và giảm
Clear	xóa TCNT2 ( cài đặt tất cả các bit về 0)
Clk <sub>Tn</sub>	xung nhịp Timer/Counter , tham khảo như là clk <sub>T0</sub> ở dưới
Top	tín hiệu răng TCNT2 vừa đạt giá trị cực đại
Bottom	tín hiệu răng TCNT2 vừa đạt giá trị cực tiêu (0)

Sự phụ thuộc của chế độ điều khiển được sử dụng, bộ đếm được xóa , làm tăng , hoặc làm giảm tại mỗi chu kì xung nhịp timer (clk<sub>T2</sub>) . clk<sub>T2</sub> có thể được sinh ra từ một nguồn xung nhịp ngoài và bên trong , được lựa chọn bằng các bit lựa chọn xung nhịp (CS22:0) . Khi không có nguồn xung nhịp nào được lựa chọn (CS22:0 = 0 ) Timer bị dừng lại . Tuy nhiên , giá trị của TCNT2 có thể được truy nhập bằng CPU , bất chấp việc clk<sub>T2</sub> được đưa ra hay không . Một CPU ghi đè lên (có quyền ưu tiên cao hơn ) tất cả các bộ đếm xóa hoặc điều khiển quá trình đếm .

Chuỗi đếm thì được xác định bằng việc cài đặt của các bit WGM01:0 đặt trong thanh ghi điều khiển Timer/Counter (TCCR2) . Có nhiều kết nối đóng giữu cách mà bộ đếm được xử lí và cách mà dạng sóng được sinh ra trên đầu ra so sánh đầu ra OC2 . để biết thêm chi tiết 1 cách hoàn thiện về các chuỗi đếm và các bộ phát dạng sóng , xem “các chế độ điều khiển ‘ trên trang 150

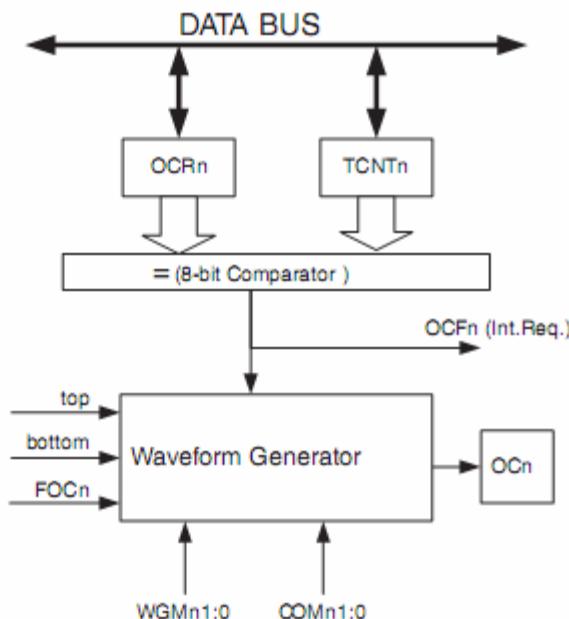
Cờ báo tràn Timer/Counter (TOV2) được cài đặt lên chế độ điều khiển được lựa chọn bằng các bit WGM2:0 . TOV2 có thể được sử dụng cho việc phát ra 1 ngắt CPU

## Bộ phận so sánh đầu ra

Bộ so sánh 8bit so sánh 1 cách liên tục TCNT2 với thanh ghi so sánh đầu ra (OCR2) . Ở bất cứ đâu TCNT2 bằng OCR2 , các tín hiệu so sánh có 1 sự tương ứng . Một sự thích ứng sẽ cài đặt cờ báo so sánh đầu ra (OCF2) tại chu kì xung nhịp tiếp theo

. Nếu được kích hoạt ( $OCIE2=1$  và các cờ ngắt chung trong thanh ghi SREG được cài đặt) , cờ báo so sánh đầu ra phát ra 1 ngắt so sánh đầu ra . Cờ báo  $OCF2$  tự động bị xóa khi mà ngắt được thực thi . Như một sự lựa chọn , cờ  $OCF2$  có thể bị xóa bằng phần mềm bởi việc viết 1 mức logic 1 lên vùng bit I/O của chúng . Máy phát dạng sóng sử dụng các tín hiệu tương ứng để phát 1 đầu ra theo chế độ điều khiển được cài bằng các bit  $WGM2:1$  và các bit chế độ đầu ra so sánh ( $COM21:0$ ). Các tín hiệu max và bottom thì được sử dụng bởi máy phát dạng sóng cho việc điều khiển trong các trường hợp đặc biệt của các giá trị cực biên trong một vài chế độ của quá trình điều khiển (xem bảng Các chế độ điều khiển trên trang 150) Hình 63 chỉ ra sơ đồ khối của bộ phận so sánh đầu ra .

**Figure 63.** Output Compare Unit, Block Diagram



Thanh ghi  $OCR2$  được ghi vào bộ đếm kép khi sử dụng bất cứ chế độ điều chế độ rộng xung nào (PWM) . Về các chế độ bình thường và chế độ điều khiển CTC , bộ đếm kép được vô hiệu hóa . Bộ đếm kép đồng bộ hóa việc cập nhật thanh ghi so sánh  $OCR2$  lên 1 trong 2 chuỗi đếm TOP và BOTTOM . Sự đồng bộ hóa này ngăn cản sự xuất hiện của odd-length , các xung PWM không đối xứng , do vậy làm cho đầu ra không có nhiễu sọc ngang .

Sự truy nhập thanh ghi  $OCR2$  có thể dường như khá phức tạp , nhưng không phải trong trường hợp này . Khi mà bộ đếm kép được kích hoạt , CPU vừa truy nhập vào bộ đếm của thanh ghi  $OCR2$  , và nếu bộ đếm kép bị vô hiệu hóa CPU sẽ truy nhập  $OCR2$  một cách trực tiếp .

### So sánh đầu ra cường bức

Trong các chế độ phát dạng sóng không phải PWM , đầu ra tương ứng của bộ so sánh có thể bị cường bức bởi việc viết 1 mức logic 1 lên bit so sánh đầu ra cường bức ( $FOC2$ ) . Ghép so sánh đầu ra cường bức này sẽ không cài đặt cờ  $OCF2$  hoặc

reload/clear timer , nhưng chân OC2 sẽ được cập nhật nếu như 1 ghép so sánh đã được xuất hiện ( việc cài đặt các bit COM21:0 xác định trạng thái chân OC2 được cài đặt , bị xóa hoặc di chuyển )

## Sự khóa ghép so sánh bằng việc viết TCNT2

Tất cả các quá trình viết lên thanh ghi TCNT2 sẽ khóa bất kì ghép so sánh nào cái mà xuất hiện trong chu kỳ xung nhịp tới , dù khi timer bị dừng lại . Đặc điểm này cho phép OCR2 được khởi tạo về giá trị giống như TCNT2 mà không khởi động 1 ngắt khi xung nhịp Timer/Counter được kích hoạt

### Việc sử dụng bộ phận so sánh đầu ra

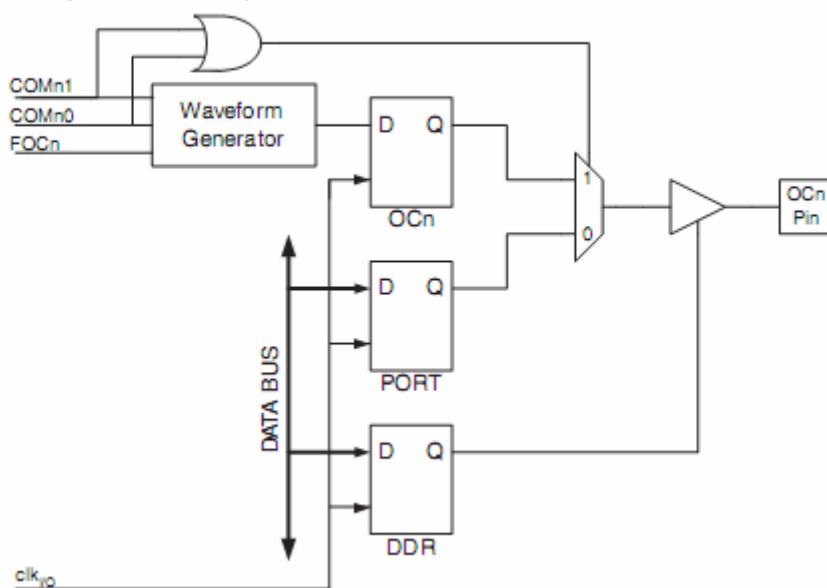
Do việc viết TCNT2 trong bất cứ chế độ điều khiển nào sẽ khóa tất cả các ghép so sánh cho 1 chu kỳ xung nhịp , có nhiều nguy cơ được kéo theo khi thay đổi TCNT2 khi sử dụng kênh so sánh đầu ra, sự phụ thuộc của trạng thái Timer/Counter chạy hoặc không . Nếu giá trị được viết lên TCNT2 bằng giá trị OCR2 , ghép so sánh sẽ bị lỗi , kết quả là quá trình phát dạng sóng không chính xác . Một cách tương tự , không viết giá trị TCNT2 bằng giá trị BOTTOM khi mà bộ đếm đang đếm xuống

Việc cài đặt OC2 nên được tiến hành trước việc cài đặt thanh ghi định hướng dữ liệu cho chân công lên công ra . Cách dễ nhất để cài đặt giá trị OC2 là sử dụng các bit đầu dò so sánh đầu ra cưỡng bức trong chế độ bình thường . Thanh ghi OC2 giữ giá trị của nó dù khi thay đổi giữa các chế độ phát dạng sóng

Để nhận thấy rằng các bit COM21:0 thì không được ghi vào bộ đếm cùng nhau với giá trị so sánh . Việc thay đổi các bit COM21:0 sẽ tạo ra ảnh hưởng ngay lập tức

### Bộ phận đầu ra ghép so sánh

Các bit chế độ đầu ra so sánh (COM21:0) có 2 chức năng . Bộ phát dạng sóng sử dụng các bit COM21:0 cho việc xác định trạng thái đầu ra so sánh (OC2) tại ghép so sánh tiếp theo . Các bit COM21:0 cũng điều khiển chân nguồn đầu ra OC2 . Hình 64 chỉ ra một sơ đồ logic đã rút gọn của logic hỏng bằng việc cài đặt COM21:0 . Các thanh ghi I/O , các bit I/O , và các chân I/O trong hình thì được in đậm . Chỉ có các phần của các thanh ghi điều khiển công I/O chung (DDR và PORT) cái mà bị ảnh hưởng bởi các bit COM21:0 là được chỉ ra . Khi tham khảo lên các trạng thái của OC2 , sự tham khảo là cho các thanh ghi OC2 bên trong , không phải các chân OC2 . Nếu tín hiệu Reset hệ thống xuất hiện , thanh ghi OC2 được Reset về 0

**Figure 64. Compare Match Output Unit, Schematic**

Các cổng I/O chức năng chung được ghi đè bằng so sánh đầu ra (OC2) từ máy phát dạng sóng nếu như 1 trong số các bit COM21:0 được cài đặt . Tuy nhiên , chân định hướng OC2 ( đầu ra hoặc đầu vào ) thì vẫn được điều khiển bằng thanh ghi định hướng dữ liệu (DDR) cho chân cổng . Bit thanh ghi định hướng cho dữ liệu cho chân OC2 (DDR\_OC2) phải được cài đặt như là đầu trước khi giá trị của OC2 được nhìn thấy trên cổng . Chức năng cổng ghi đè thì độc lập trong chế độ phát dạng sóng .

Thiết kế của chân logic so sánh đầu ra cho phép việc khởi tạo của trạng thái OC2 trước khi đầu ra được kích hoạt . Chú ý rằng vài việc cài đặt các bit COM21:0 là dự trữ cho các chế độ điều khiển nào đó . Xem bảng miêu tả thanh ghi của Timer/Counter 8bit trên trang 157

### Sự phát dạng sóng và chế độ đầu ra so sánh

Máy phát dạng sóng sử dụng các bit CON21:0 theo các cách khác nhau trong chế độ bình thường , CTC , và các chế độ PWM. Cho tất cả các chế độ , việc cài đặt COM21:0 nói cho máy phát dạng sóng rằng không có hành động nào trên thanh ghi OC2 được tiến hành trên ghép so sánh tiếp theo . Về các đầu ra so sánh hoạt động trong các chế độ không phải là PWM tham khảo bảng 65 trên trang 158 . Về các chế độ PWM , tham khảo bảng 66 trang 158, và các chế độ PWM đúng pha tham khảo bảng 67 trên trang 158 .

Một sự thay đổi trạng thái của các bit COM21:0 sẽ có hiệu lực tại ghép so sánh đầu tiên sau khi các bit được ghi . Về các chế độ không phải PWM , các hoạt động có thể bị cưỡng bức để có hiệu lực ngay lập tức bằng việc sử dụng các bit phân tích FOC2

## Các chế độ của quá trình điều khiển

Các chế độ điều khiển , ví dụ như quá trình xử lí của Timer/Counter và các chân so sánh đầu ra , được xác định bằng việc kết hợp của chế độ Phát dạng sóng (WGM21:0) và các bit chế độ so sánh đầu ra (COM21:0) . Các bit Chế độ đầu ra so sánh thì không ảnh hưởng đến các chuỗi đếm trong khi các bít chế độ phát dạng sóng thì có ảnh hưởng . Các bit COM21:0 điều khiển đầu ra của PWM là đảo hay không đảo (PWM đảo hay là không đảo ) . Về các chế độ không phải là PWM các bit COM21:0 điều khiển đầu ra nên được cài đặt , được xóa , hoặc di chuyển tại 1 ghép so sánh (xem phần “Compare Match Output Unit “ trên trang 149 .

Để biết thêm chi tiết về thông tin bộ định thời tham khảo hình 68 , hình 69 , hình 70 , hình 71 trong “các sơ đồ bộ định thời Timer/Counter” trang 155

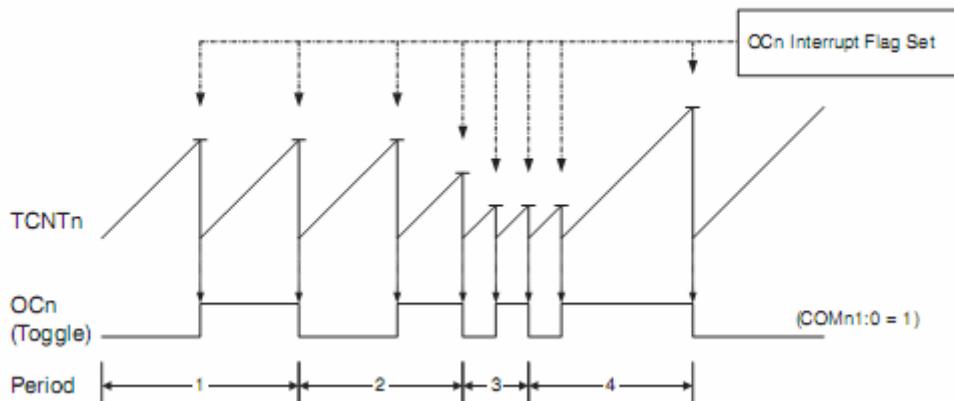
### Chế độ bình thường

Chế độ đơn giản của quá trình điều khiển là chế độ normal (WGM21:0=0) . Trong chế độ này việc định hướng đếm thì luôn luôn lên trên (tăng dần) , và không có xóa bộ đếm nào được tiến hành . Bộ đếm đơn giản tràn khi nó vượt qua giá trị 8bit max (TOP=0xFF) và sau đó khởi động lại từ giá trị bottom (0x00) . Trong quá trình điều khiển thông thường,cờ báo tràn Timer/Counter (TOV2) sẽ được cài đặt trong cùng 1 chu kì xung nhịp timer như là TCNT2 trở về 0 . Cờ báo TOV2 trong trường hợp này hành động như 1 bit thứ 9 , ngoại trừ việc nó chỉ được cài đặt , không bị xóa . Tuy nhiên , được kết nối với ngắt tràn Timer cái mà tự động xóa cờ TOV2 , độ chính xác của Timer có thể được tăng lên bằng phần mềm . Không có trường hợp đặc biệt nào được xét đến trong chế độ bình thường , 1 giá trị bộ đếm mới có thể được viết bất cứ thời gian nào

### Clear Timer on Compare Match (CTC) Mode

Trong chế độ CTC (WGM21:0=2) , thanh ghi OCR2 được sử dụng để điều khiển độ chính xác của bộ đếm . Trong chế độ CTC bộ đếm bị xóa về 0 khi mà giá trị bộ đếm (TCNT2) tương ứng với chân OC2 . Chân OC2 xác định giá trị TOP cho bộ đếm , do đó cũng thay đổi độ chính xác của nó . Chế độ này cho phép điều khiển tốt hơn tần số đầu ra của ghép so sánh . Nó cũng làm đơn giản quá trình điều khiển của việc đếm các biến có bên ngoài .

Giản đồ thời gian của chế độ CTC được chỉ ra trong hình 65 . Giá trị của bộ đếm (TCNT2) tăng lên cho đến khi 1 ghép so sánh xuất hiện giữa TCNT2 và OCR2 và sau đó bộ đếm (TCNT2 ) bị xóa

**Figure 65. CTC Mode, Timing Diagram**

Một ngắt có thể được sinh ra mỗi lần mà giá trị bộ đếm tiến đến giá trị TOP bằng việc sử dụng cờ OCF2 . Nếu ngắt được kích hoạt , các chương trình con điều khiển ngắt có thể được sử dụng cho việc cập nhật giá trị TOP . Tuy nhiên , việc thay đổi TOP tới 1 giá trị ẩn lên BOTTOM khi mà bộ đếm đang chạy với giá trị bộ đếm gộp trước thấp hoặc không có thì phải được thực hiện 1 cách thận trọng từ đó mà chế độ CTC không có đặc tính bộ đếm kép . Nếu giá trị mới được ghi lên OCR2 thấp hơn giá trị TCNT2 hiện thời , bộ đếm sẽ bị lỗi ghép so sánh . Bộ đếm sau đó sẽ phải đếm đến giá trị cực đại của nó (0xFF) và bao quanh giá trị bắt đầu tại 0x00 trước khi ghép so sánh có thể xuất hiện .

việc phát ra một đầu ra dạng sóng trong chế độ CTC , đầu ra OC2 có thể được cài đặt để di chuyển mức logic trên mỗi ghép so sánh bằng việc cài đặt các bit chế độ đầu ra so sánh lên chế độ di chuyển (COM21:0=1). Giá trị OC2 sẽ không thể nhìn thấy trên chân cổng trừ khi sự định hướng dữ liệu cho chân được cài đặt lên đầu ra . Dạng sóng được phát ra sẽ có giá trị tần số lớn nhất của  $f_{OC2} = f_{clk\_I/O}/2$  khi mà OCR2 được cài đặt về 0(0x00) . Tần số sóng phát ra được xác định bằng công thức dưới đây

$$f_{OCn} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRn)}$$

Biến N được đưa ra theo tỉ lệ (1 , 8 , 64, 256 , 1024)

Như là trong chế độ normal , cờ báo TOV2 được cài đặt trong cùng một chu kì xung nhịp timer cái mà bộ đếm đếm từ giá trị MAX về 0x00

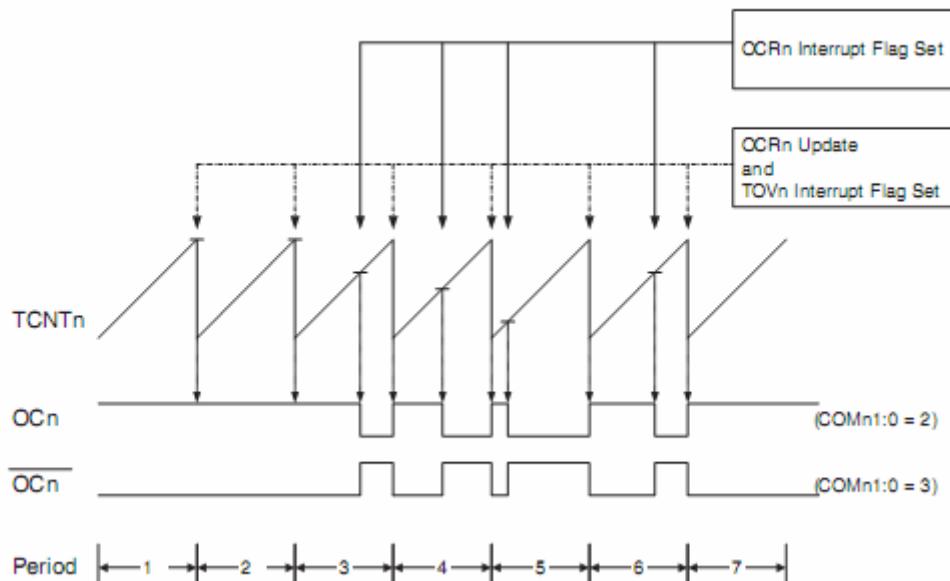
## Chế độ fast PWM

Chế độ điều chế độ rộng xung nhanh hay fast PWM (WGM21:0=3) cung cấp một lựa chọn phát dạng sóng PWM tần số cao . Chế độ fast PWM khác với các chế độ PWM khác bởi quá trình điều khiển sùm đơn của nó . Bộ đếm đếm từ BOTTOM đến MAX sau đó bắt đầu lại từ chế BOTTOM . Trong chế độ đầu ra so sánh không đảo , đầu ra so sánh (OC2) bị xóa trên ghép so sánh giữa TCNT2 và OCR2 , và được cài đặt ở BOTTOM . Trong chế độ đầu ra so sánh đảo , đầu ra được cài đặt trên ghép so sánh và bị xóa ở BOTTOM . Dù cho chế độ điều khiển sùm đơn , tần số điều khiển

của chế độ fast PWM có thể cao gấp đôi chế độ PWM đúng pha sử dụng chế độ điều khiển 2 sườn . Tần số cao này làm cho chế độ fast PWM phù hợp với sự điều chỉnh nguồn điện , sự chỉnh lưu và các ứng dụng DAC . Tần số cao cho phép các thành phần vật lí cõi ngoài (dây cuộn , tụ điện ) , và vì vậy giảm giá thành hệ thống

Trong chế độ fast PWM , bộ đếm được làm tăng cho đến khi giá trị của bộ đếm tương ứng với giá trị max. Bộ đếm sau đó được xóa tại chu kỳ xung nhịp kế tiếp . Giản đồ thời gian cho chế độ fast PWM được chỉ ra như trong hình 66 . Giá trị của TCNT2 trong giản đồ thời gian được chỉ ra như 1 biểu đồ minh họa cho quá trình điều khiển sườn đơn . Giản đồ bao gồm các đầu ra PWM đảo và không đảo . Đường thẳng đứng nhỏ đánh dấu trên sườn TCNT2 đưa ra các ghép so sánh giữa OCR2 và TCNT2

**Figure 66. Fast PWM Mode, Timing Diagram**



Cờ báo tràn Timer/Counter (TOV2) thì được cài đặt mỗi lần mà bộ đếm đạt tới giá trị Max nếu ngắt được kích hoạt , các chương trình con điều khiển ngắt có thể được sử dụng cho việc cập nhật các giá trị mới .

Trong chế độ fast PWM , bộ phận so sánh cho phép quá trình phát ra các dạng sóng PWM trên chân OC2 . Việc cài đặt các bit COM21:0 lên 2 sẽ gây ra 1 đầu ra PWM không đảo và 1 đầu ra PWM đảo có thể được sinh ra bằng việc cài đặt các bit COM21:0 lên 3 (xem bảng 66 trang 158) Giá trị thật của OC2 sẽ chỉ được nhìn thấy trên chân công nếu sự định hướng dữ liệu cho các chân công được cài đặt như là đầu ra . Dạng sóng PWM được sinh ra bằng việc cài đặt (hoặc xóa ) thanh ghi OC2 tại chu kỳ xung nhịp mà bộ đếm bị xóa (thay đổi từ MAX đến BOTTOM)

Tần số PWM cho đầu ra có thể được tính toán bằng công thức dưới đây :

$$f_{OCnPWM} = \frac{f_{clk\_I/O}}{N \cdot 256}$$

Giá trị của biến N theo tỉ lệ xích sau (1, 8, 64 , 256 , 1024)

Giá trị cực biên cho thanh ghi OCR2 được đưa ra trong các trường hợp đặc biệt khi mà quá trình 1 dạng sóng đầu ra PWM trong chế độ fast PWM . Nếu OCR2 được cài đặt bằng BOTTOM , đầu ra sẽ là 1 định hép cho mỗi chu kỳ xung nhịp MAX+1 .

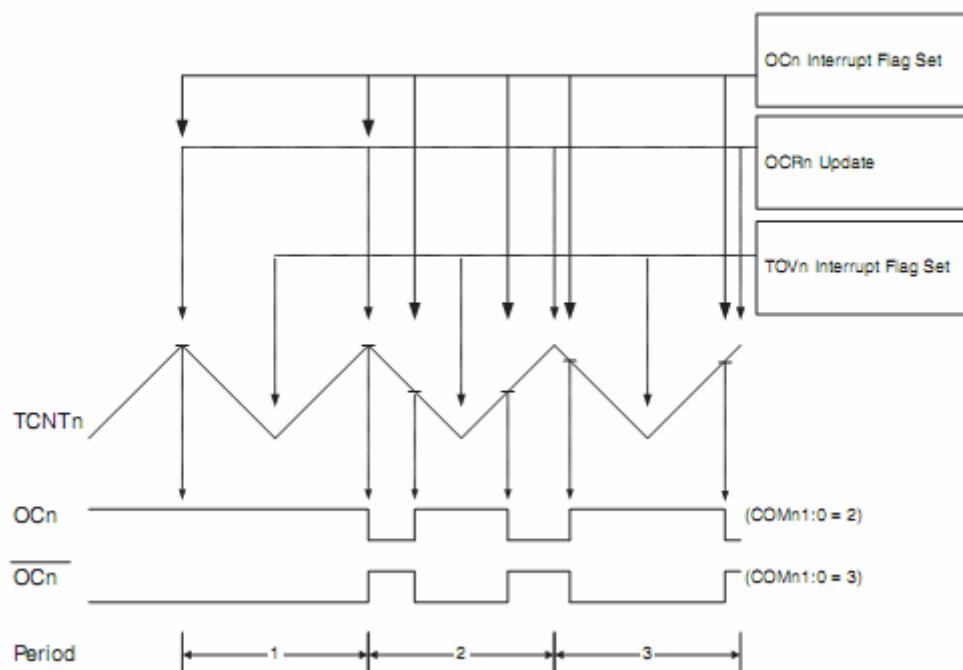
Việc cài đặt OCR2 bằng giá trị MAX sẽ cho kết quả là một giá trị cao không đổi hoặc đầu ra thấp (phụ thuộc vào độ phân cực của đầu ra được cài đặt bằng các bit COM21:0)

Một tần số (với 50% chu kỳ tải) dạng sóng đầu ra trong chế độ fast PWM có thể đạt được bằng việc cài đặt OC2 lên cấp logic của nó di chuyển trên mỗi cấp so sánh (COM21:0 = 1). Dạng sóng được sinh ra sẽ có tần số max của  $f_{OC2} = f_{clk\_I/O}/2$  khi mà OCR2 được cài đặt là 0. Đặc điểm này thì tương tự như sự di chuyển OC2 trong chế độ CTC mode, ngoại trừ đặc điểm bộ đếm kép của bộ phận đầu ra so sánh được kích hoạt trong chế độ fast PWM

## Chế độ PWM đúng pha

Chế độ PWM đúng pha (WGM21:0 = 1) cung cấp một độ chính xác cao lựa chọn phát dạng sóng PWM đúng pha. Chế độ PWM đúng pha thì cơ bản dựa trên quá trình điều khiển 2 sườn. Bộ đếm đếm lặp lại từ BOTTOM đến MAX và sau đó từ MAX đến BOTTOM. Trong chế độ đầu ra so sánh không đảo, đầu ra so sánh (OC2) bị xóa trên ghép so sánh giữa TCNT2 và OCR2 trong khi đang đếm lên. và cài đặt trên ghép so sánh trong khi đang đếm xuống. Trong chế độ so sánh đầu ra đảo, quá trình điều khiển này bị đảo ngược. Chế độ điều khiển 2 sườn có tần số cực đại của quá trình điều khiển thấp hơn chế độ điều khiển sườn đơn. Tuy nhiên, dù cho đặc tính đổi xung của các chế độ PWM 2 sườn, các chế độ này được ưa thích hơn cho các ứng dụng điều khiển động cơ

Độ chính xác của PWM cho chế độ PWM đúng pha là cố định 8bit. Trong chế độ PWM đúng pha, bộ đếm được làm tăng cho đến khi giá trị bộ đếm max khi mà bộ đếm đạt tới giá trị MAX, nó thay đổi hướng đếm. Giá trị TCNT2 sẽ bằng giá trị MAX trong mỗi chu kỳ xung nhịp timer. Giản đồ thời gian cho chế độ PWM đúng pha được chỉ ra trong hình 67. Giá trị TCNT2 trong giản đồ thời gian chỉ ra 1 biểu đồ để minh họa cho quá trình điều khiển sườn kép. Biểu đồ bao gồm các đầu ra PWM đảo và không đảo... Đường thẳng đứng nhỏ đánh dấu trên sườn TCNT2 đưa ra các ghép so sánh giữa OCR2 và TCNT2

**Figure 67.** Phase Correct PWM Mode, Timing Diagram

Cờ báo tràn Timer/Counter (TOV2) được cài đặt mỗi lần bộ đếm vươn tới giá trị BOTTOM . Cờ ngắt có thể được sử dụng để phát ra một ngắt mỗi lần bộ đếm đạt tới giá trị BOTTOM.

Trong chế độ PWM đúng pha , bộ phận so sánh cho phép quá trình phát ra các dạng sóng PWM trên chân OC2 . Việc cài đặt các bit COM21:0 lên 2 sẽ gây ra 1 đầu ra PWM không đảo và 1 đầu ra PWM đảo có thể được sinh ra bằng việc cài đặt các bit COM21:0 lên 3 (xem bảng 66 trang 158) Giá trị thật của OC2 sẽ chỉ được nhìn thấy trên chân cổng nếu sự định hướng dữ liệu cho các chân cổng được cài đặt như là đầu ra . Dạng sóng PWM được sinh ra bằng việc cài đặt (hoặc xóa) thanh ghi OC2 tại ghép so sánh giữa OCR2 và TCNT2 khi mà bộ đếm tăng lên , và việc cài đặt (hoặc xóa) thanh ghi OC2 tại ghép so sánh giữa OCR2 và TCNT2 khi mà bộ đếm giảm . Tần số PWM cho đầu ra khi sử dụng chế độ PWM đúng pha có thể được tính toán bằng công thức dưới đây

$$f_{OC_n PC PWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

Giá trị của biến N theo tỉ lệ xích sau (1, 8, 64 , 256 , 1024)

Giá trị cực biên của thanh ghi OCR2 đưa ra trong các trường hợp đặc biệt khi mà việc phát ra các đầu ra dạng sóng trong chế độ PWM đúng pha . Nếu OCR2 được cài đặt bằng BOTTOM, đầu ra sẽ liên tiếp ở mức thấp và nếu cài đặt bằng MAX , đầu ra sẽ tiếp tục ở mức cao trong chế độ PWM không đảo . Về các chế độ PWM đảo đầu ra sẽ có giá trị đổi lặp lại .

Tại mỗi điểm bắt đầu của chu kỳ 2 trong hình 67 Ocn có 1 sự chuyển đổi từ mức cao xuống mức thấp dù cho không có ghép so sánh nào . Điểm đánh dấu sự chuyển đổi

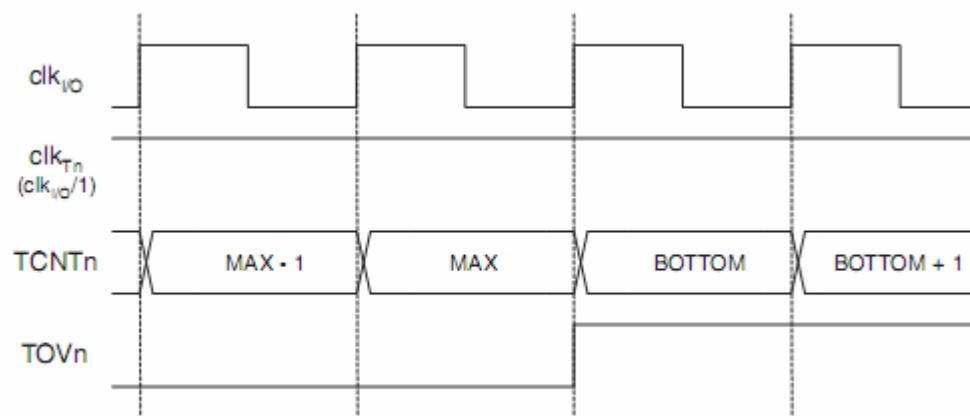
này để đảm bảo tính đối xứng xung quanh BOTTOM . Có 2 trường hợp có sự chuyển đổi mà không có ghép so sánh :

- OCR2A chuyển giá trị của nó từ MAX , giống như hình 67 . Khi giá trị của OCR2A là MAX , giá trị chân OCn thì giống như kết quả của 1 ghép so sánh đếm xuống . Để đảm bảo tính đối xứng xung quanh BOTTOM , giá trị của OCn tại MAX phải tương ứng với kết quả của một ghép so sánh đếm lên .
- Timer bắt đầu đếm từ 1 giá trị cao hơn 1 trong OCR2A , và vì lí do này mà các lỗi của ghép do sánh và do đó OCn thay đổi cách mà nó xuất hiện trên đường lên .

## Giản đồ thời gian của Timer/Counter

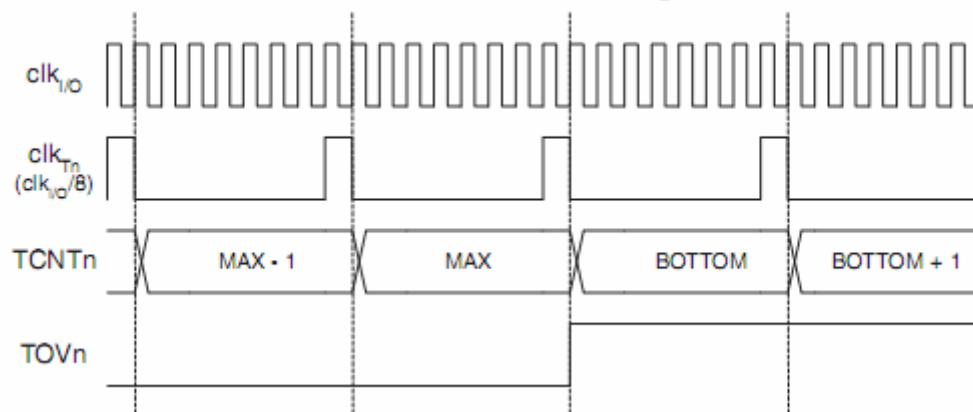
Timer/Counter là một thiết kế đồng bộ và xung nhịp timer ( $\text{clk}_{\text{T2}}$ ) vì vậy được chỉ ra như là 1 tín hiệu kích hoạt xung nhịp trong các hình dưới đây . Các hình bao gồm các dữ kiện khi mà các cờ ngắt được cài đặt . Hình 88 bao gồm dữ liệu thời gian cho quá trình điều khiển Timer/Counter cơ bản . Hình chỉ ra các chuỗi đếm đóng đến các giá trị MAX trong tất cả các chế độ PWM đúng pha

**Figure 68.** Timer/Counter Timing Diagram, no Prescaling



**Figure 69** shows the same timing data, but with the prescaler enabled.

**Figure 69.** Timer/Counter Timing Diagram, with Prescaler ( $f_{\text{clk\_I/O}}/8$ )



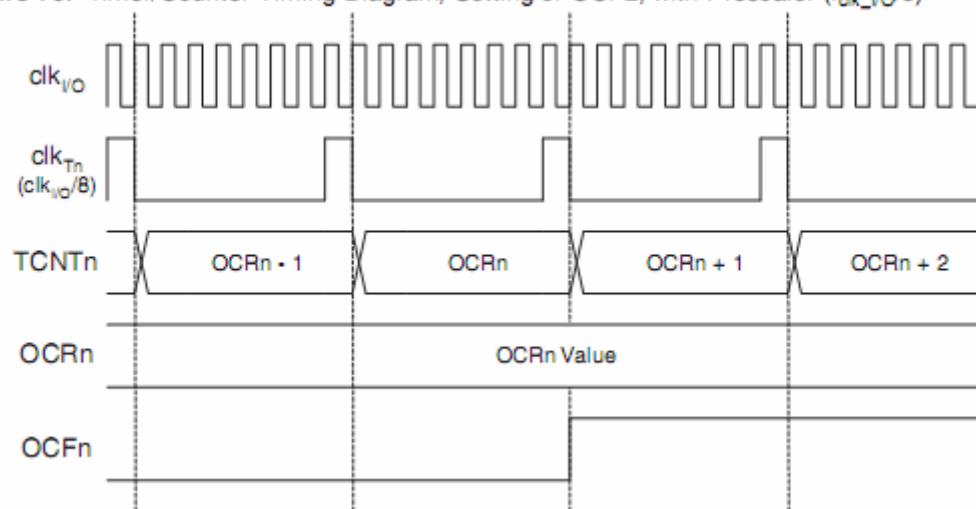
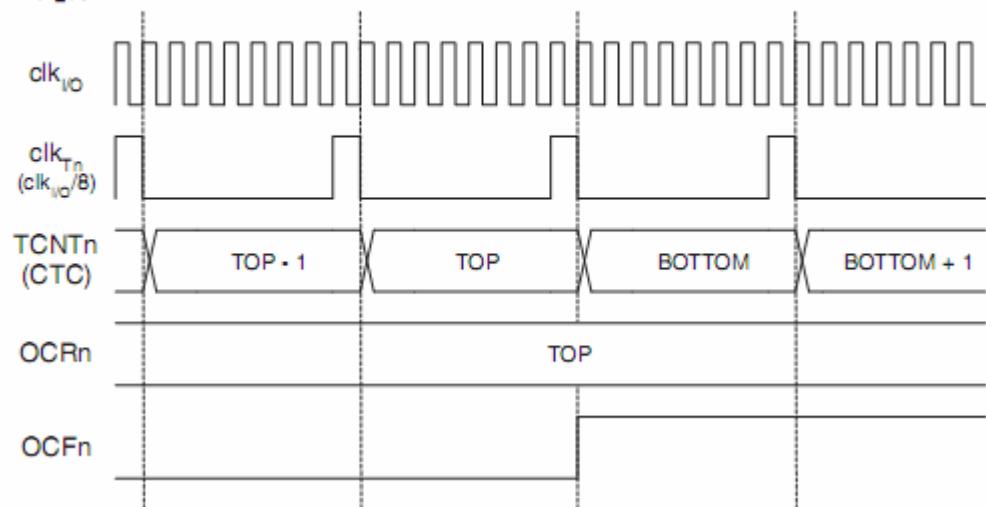
**Figure 70.** Timer/Counter Timing Diagram, Setting of OCF2, with Prescaler ( $f_{clk_{IO}}/8$ )

Figure 71 shows the setting of OCF2 and the clearing of TCNT2 in CTC mode.

**Figure 71.** Timer/Counter Timing Diagram, Clear Timer on Compare Match Mode, with Prescaler ( $f_{clk_{IO}}/8$ )

## Sự miêu tả thanh ghi Timer/Counter 8bit

### Thanh ghi điều khiển Timer/Counter – TCCR2

Bit	7	6	5	4	3	2	1	0	TCCR2
Read/Write	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	
Initial Value	W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Bit 7 – FOC2 : so sánh đầu ra cưỡng bức

Bit FOC2 chỉ hoạt động khi bit WGM20 xác định chế độ không phải PWM . Tuy nhiên , để đảm bảo tính tương thích với các thiết bị trong tương lai , bit này phải được cài đặt là 0 khi mà TCCR2 được ghi khi hoạt động trong chế độ PWM . Khi viết 1 mức logic 1 lên bit FOC2 , 1 ghép so sánh ngay trung gian bị cưỡng bức trên bộ phận phát dạng sóng . Đầu ra OC2 bị thay đổi theo việc cài đặt các bit COM21:0 của nó . Chú ý rằng bit FOC2 thì thực hiện như là 1 đầu dò(strobe) . Vì vậy nó là giá trị được đưa ra trong các bit COM21:0 xác định ảnh hưởng của so sánh cưỡng bức

Một đầu dò FOC2 sẽ không phát ra bất cứ ngắt nào , cũng sẽ không xóa Timer trong chế độ CTC sử dụng OCR2 như là già trị TOP.

Bit FOC2 thì luôn được đọc là 0

Bit 6,3 – WGM21:0 : chế độ phát dạng sóng

Các bit này điều khiển chuỗi đếm của bộ đếm , nguồn cho giá trị đếm cực đại TOP , và loại mà phát dạng sóng được sử dụng . Các chế độ điều khiển hỗ trợ bởi bộ phận Timer/Counter là : Chế độ Normal , chế độ CTC , và 2 loại của các chế độ điều chế độ rộng xung PWM , xem bảng 64 và “các chế độ điều khiển “ trang 150

Table 64. Waveform Generation Mode Bit Description

Mode	WGM21 (CTC2)	WGM20 (PWM2)	Timer/Counter Mode of Operation	TOP	Update of OCR2 at	TOV2 Flag Set on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR2	Immediate	MAX
3	1	1	Fast PWM	0xFF	BOTTOM	MAX

Note: The CTC2 and PWM2 bit definition names are now obsolete. Use the WGM21:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

Bit 5:4 – COM21:0 : chế độ đầu ra ghép so sánh

Các bit này điều khiển chân xử lí so sánh đầu ra (OC2) . Nếu 1 trong 2 bit COM21:0 được cài đặt , đầu ra OC2 ghi đè lên công chức năng thông thường của chân I/O mà nó được nối đến . Tuy nhiên , chú ý rằng bit thanh ghi định hướng dữ liệu (DDR) tương ứng đến chân OC2 phải được cài đặt theo thứ tự kích hoạt bộ điều khiển đầu ra

Khi OC2 được nối với chân , chức năng của các bit COM21:0 phụ thuộc vào việc cài đặt bit WGM21:0 . Bảng 65 chỉ ra chức năng của bit COM21:0 khi các bit WGM21:0 được cài đặt đến chế độ bình thường hoặc chế độ CTC (không phải PWM)

**Table 65.** Compare Output Mode, Non-PWM Mode

COM21	COM20	Description
0	0	Normal port operation, OC2 disconnected.
0	1	Toggle OC2 on compare match
1	0	Clear OC2 on compare match
1	1	Set OC2 on compare match

Bảng 66 chỉ ra các bit chức năng COM21:0 khi mà các bit WGM21:0 được cài đặt trong chế độ fast PWM

**Table 66.** Compare Output Mode, Fast PWM Mode<sup>(1)</sup>

COM21	COM20	Description
0	0	Normal port operation, OC2 disconnected.
0	1	Reserved
1	0	Clear OC2 on compare match, set OC2 at BOTTOM, (non-inverting mode)
1	1	Set OC2 on compare match, clear OC2 at BOTTOM, (inverting mode)

Note: 1. A special case occurs when OCR2 equals TOP and COM21 is set. In this case, the compare match is ignored, but the set or clear is done at BOTTOM. See "Fast PWM Mode" on page 152 for more details.

Bảng 67 chỉ ra chức năng của bit COM21:0 khi mà các bit WGM21:0 được cài đặt trong chế độ PWM đúng pha

**Table 67.** Compare Output Mode, Phase Correct PWM Mode<sup>(1)</sup>

COM21	COM20	Description
0	0	Normal port operation, OC2 disconnected.
0	1	Reserved
1	0	Clear OC2 on compare match when up-counting. Set OC2 on compare match when downcounting.
1	1	Set OC2 on compare match when up-counting. Clear OC2 on compare match when downcounting.

Note: 1. A special case occurs when OCR2 equals TOP and COM21 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 153 for more details.

Bit 2:0 – CS22 : lựa chọn xung nhịp

Có 3 bit lựa chọn xung nhịp lựa chọn nguồn xung nhịp sẽ được sử dụng bởi Timer/Counter

**Table 68.** Clock Select Bit Description

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	clk <sub>IO</sub> /No prescaling
0	1	0	clk <sub>IO</sub> /8 (From prescaler)
0	1	1	clk <sub>IO</sub> /64 (From prescaler)
1	0	0	clk <sub>IO</sub> /256 (From prescaler)

CS22	CS21	CS20	Description
1	0	1	clk <sub>IO</sub> /1024 (From prescaler)
1	1	0	External clock source on T2 pin. Clock on falling edge
1	1	1	External clock source on T2 pin. Clock on rising edge

Nếu các chế độ chân ngoài được sử dụng cho Timer/Counter 2 thì việc chuyển đổi trên chân T2 sẽ khóa bộ đếm dù nếu chân được cấu hình như là 1 đầu ra . Đặc điểm này cho phép phần mềm điều khiển việc đếm

### Thanh ghi Timer/Counter – TCNT2

Bit	7	6	5	4	3	2	1	0	TCNT2
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi Timer/Counter đưa ra sự truy cập trực tiếp , cả hai quá trình đọc và ghi lên bộ đếm Timer/Counter 8 bit . Việc viết thanh ghi TCNT2 khóa (gỡ bỏ ) ghép so sánh trên các xung nhịp kế tiếp . Quá trình sửa đổi bộ đếm (TCNT2) trong khi bộ đếm đang chạy , đưa ra 1 nguy cơ lỗi ghép so sánh giữa thanh ghi TCNT2 và OCR2

### Thanh ghi so sánh đầu ra – OCR2

Bit	7	6	5	4	3	2	1	0	OCR2
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi so sánh đầu ra bao gồm 1 giá trị 8bit cái mà được so sánh 1 cách liên tục với giá trị bộ đếm (TCNT2) . Một ghép có thể được sử dụng để sinh ra 1 ngắt so sánh đầu ra , hoặc phát ra 1 đầu ra dạng sóng trên chân OC2

### Thanh ghi che ngắt Timer/Counter - TIMSK

Bit	7	6	5	4	3	2	1	0	TIMSK
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – OCIE2 : kích hoạt ngắt ghép so sánh đầu ra Timer/Counter

Khi bit OCIE2 được ghi là 1 , và bit I trong thanh ghi trạng thái được đặt là 1 , ngắt ghép so sánh Timer/Counter 2 được kích hoạt . Ngắt tương ứng được thực thi nếu 1 ghép so sánh trong Timer/Counter xuất hiện ví dụ khi bit OCF2 được cài đặt trong thanh ghi cờ báo ngắt Timer/Counter – TIFR

Bit 6 – TOIE2 : kích hoạt ngắt tràn Timer/Counter2

Khi bit TOIE2 được ghi là 1 , và bit I trong thanh ghi trạng thái được cài đặt là 1 , ngắt tràn Timer/Counter 2 được kích hoạt . Ngắt tương ứng được thực thi nếu 1 cờ báo tràn trong Timer/Counter 2 xuất hiện ví dụ , khi bit TOV2 được cài đặt trong thanh ghi cờ báo ngắt Timer/Counter – TIFR

## Thanh ghi cờ báo ngắt Timer/Counter – TIFR

Bit	7	6	5	4	3	2	1	0	TIFR
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – OCF2 : cờ ghép so sánh 2

Bit OCF2 được cài đặt là 1 khi 1 ghép so sánh xuất hiện giữa Timer/Counter và dữ liệu trong OCR2 – thanh ghi so sánh đầu ra 2 . OCF2 bị xóa bằng phần cứng khi quá trình thực thi các vecto điều khiển ngắt tương ứng . Như một sự lựa chọn , OCF2 bị xóa bằng việc viết 1 mức logic 1 lên cờ . Khi mà bit I trong SREG , OCIE2 ( kích hoạt ngắt ghép so sánh Timer/Counter2) và OCF2 được cài đặt là 1 , ngắt ghép so sánh Timer/Counter 2 được thực thi .

Bit 6 – TOV2 : cờ báo tràn Timer/Counter 2

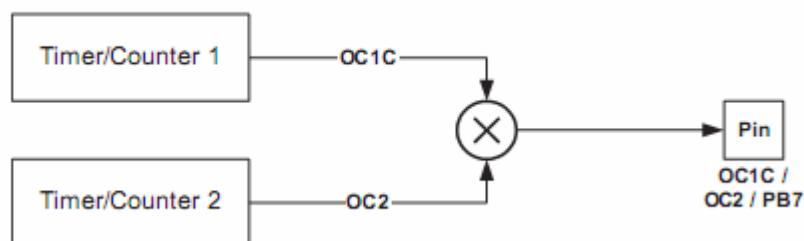
Bit TOV2 được cài đặt là 1 khi 1 sự tràn xuất hiện trong Timer/Counter 2 , TOV2 bị xóa bằng phần cứng khi mà việc thực thi các vecto điều khiển các ngắt tương ứng . Như một sự lựa chọn , TOV2 bị xóa bằng việc viết một mức logic 1 lên cờ . Khi mà bit I trong SREG , TOIE2 (kích hoạt ngắt tràn Timer/Counter 2 ) và TOV2 được cài đặt là 1 , ngắt tràn Timer/Counter 2 được thực thi . Trong chế độ PWM , bit này được cài đặt khi Timer/Counter 2 thay đổi việc đếm hướng đếm tại \$00

## XV . Khối module so sánh đầu ra (OCM1C2) – Output Compare Modulator

### Tổng quát

Module so sánh đầu ra (OCM) cho phép quá trình phát sinh của sóng điều chế với 1 tần số mang . Bộ điều chế sử dụng các đầu ra từ bộ phận so sánh đầu ra C của Timer/Counter1 -16bit và bộ phận so sánh đầu ra của Timer/Counter2- 8bit . Để biết thêm chi tiết về các Timer/Counter xem “16bit Timer/Counter (Timer/Counter 1 và Timer/Counter 3 )” ở trang 112 và Timer/Counter2 – 8bit với chế độ PWM “ ở trang 146 . Chú ý rằng đặc điểm này thì không khả dụng trong chế độ tương thích với Atmega 103 .

**Figure 72.** Output Compare Modulator, Block Diagram

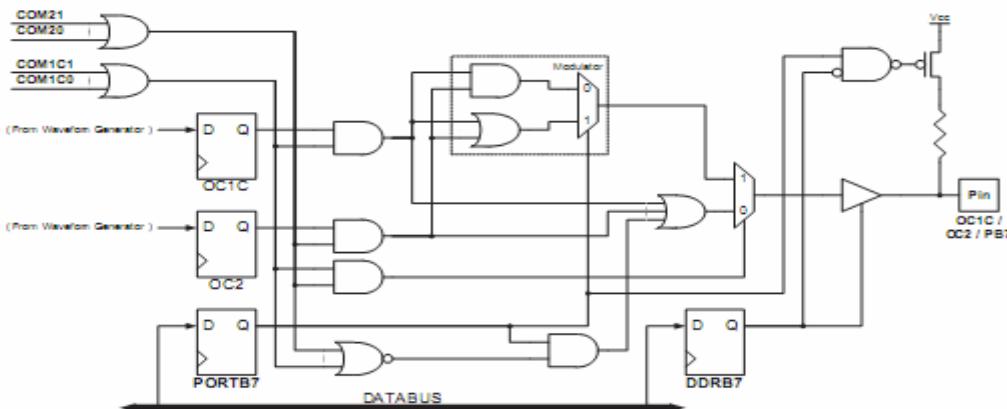


Khi bộ điều chế được kích hoạt , 2 kênh so sánh đầu ra thì được điều chế cùng nhau như được chỉ trong sơ đồ khối (hình 72)

### Miêu tả

Bộ phận so sánh đầu vào 1C và so sánh đầu vào 2 chia sẻ chân cổng PB7 như là cổng ra . Đầu ra của các bộ so sánh đầu ra (OC1C và OC2) ghi đè lên thanh ghi cổng PORTB7 thông thường khi mà 1 trong số chúng được kích hoạt (Ví dụ : khi mà bít COMnx1 : 0 không bằng 0 ) khi cả 2 bộ so sánh ngoài . OC1C và OC2 được kích hoạt trong cùng 1 thời gian , bộ điều chế sẽ tự động được kích hoạt

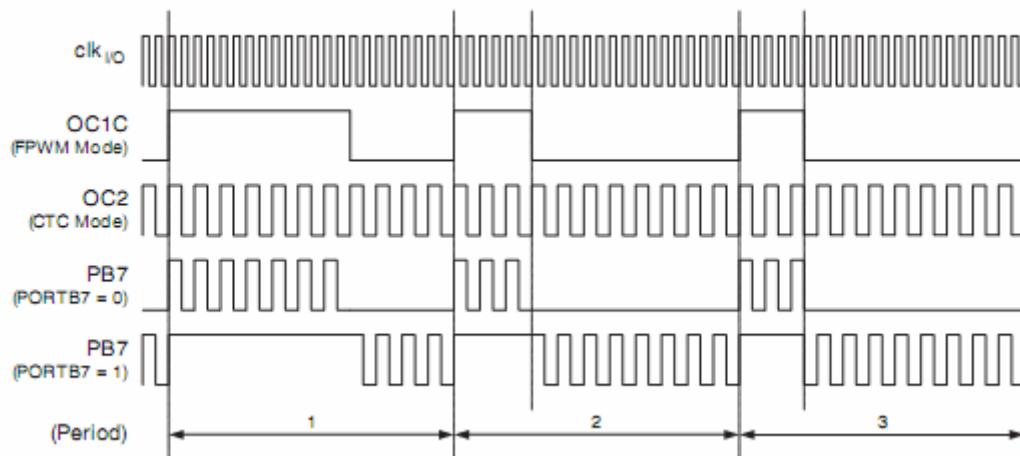
Sơ đồ chức năng tương đương của bộ điều chế được chỉ ra trong hình 73 . Sơ đồ thì bao gồm các phần của bộ Timer/Counter và mạch điều khiển đầu ra chân số 7 của cổng B

**Figure 73.** Output Compare Modulator, Schematic

Khi mà bộ điều chế được kích hoạt loại của sự điều chế (các biến logic And và Or ) có thể được lựa chọn bằng thanh ghi PORTB7 . Chú ý rằng thanh ghi DDRB7 điều khiển hướng của cổng phụ thuộc vào việc cài đặt bít COMnx1 :0

### Ví dụ về giản đồ thời gian : Timing Example

Hình 74 minh họa 1 bộ điều chế đang hoạt động . Trong ví dụ này bộ Timer/Counter 1 được cài đặt để điều khiển trong chế độ Fast PWM ( không đảo) và bộ Timer/Counter 2 sử dụng chế độ phát xung CTC với chế độ đầu ra so sánh di chuyển (Bít COMnx1 :0 = 1 )

**Figure 74.** Output Compare Modulator, Timing Diagram

Trong ví dụ này , Timer/Counter 2 cung cấp 1 sóng mang , trong khi tín hiệu điều chế được sinh ra bởi bộ phận so sánh đầu ra C của Timer/Counter 1

Độ chính xác của tín hiệu PWM (OC1C ) thì bị suy giảm bởi quá trình điều chế . Tỉ lệ suy giảm thì bằng số thứ tự của các chu kì xung nhịp hệ thống của 1 chu kì của sóng mang (OC2) .Trong chế độ này , độ chính xác bị suy giảm bởi 1 tỉ lệ với 2 . Nguyên nhân của sự suy giảm này được minh họa trong hình 74 ở chu kì thứ 2 và thứ 3 của đầu ra PB7 khi mà bít PORTB7 = 0 . Thời gian cao hơn ở chu kì thứ 2

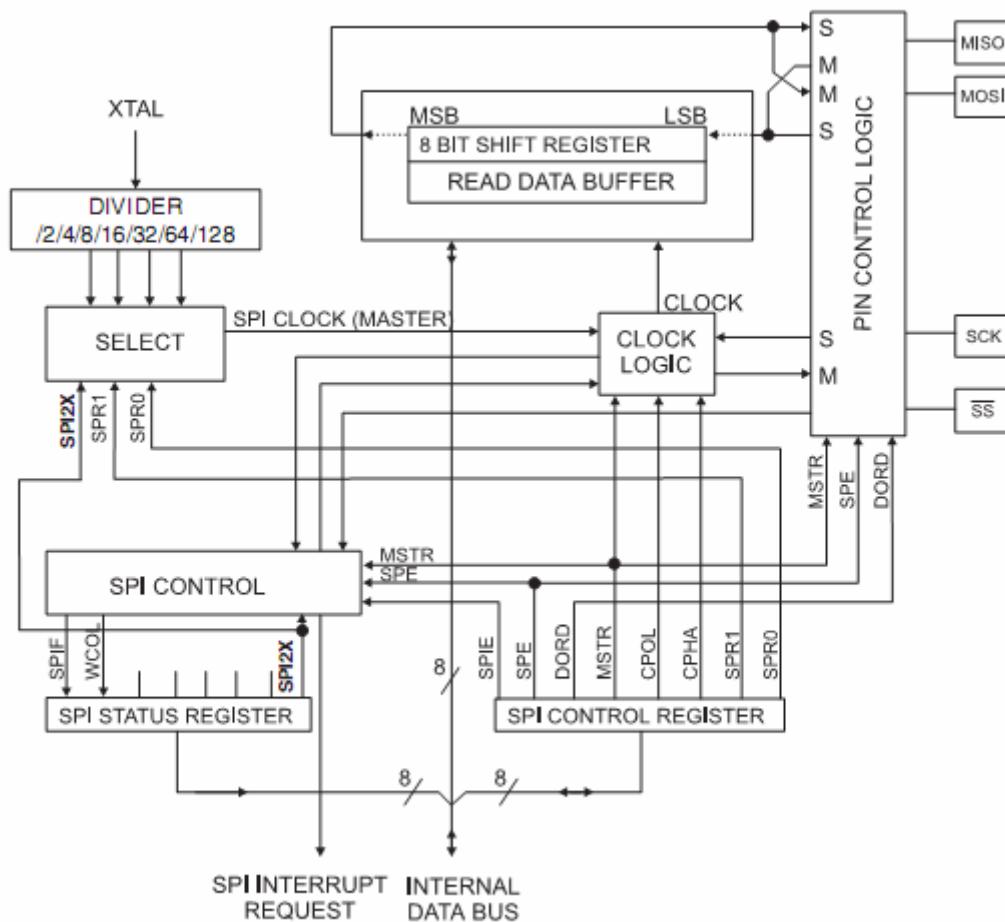
là 1 chu kì xung nhịp dài hơn trong thời gian cao của chu kì thứ 3 , nhưng kết quả thu được trên cổng ra PB7 thì bằng cả 2 chu kì đầu .

## XVI . Giao diện ngoại vi nối tiếp Serial peripheral interface – SPI

Giao diện ngoại vi nối tiếp (SPI) cho phép chuyển dữ liệu đồng bộ tốc độ cao giao Atmega128 và các thiết bị ngoại vi hoặc giữa nhiều thiết bị AVR với nhau . Giao diện ngoại vi nối tiếp (SPI) bao gồm các đặc điểm dưới đây

- Full duplex (Truyền song công ) , chuyển dữ liệu đồng bộ 3 dây (Three wire)
- Chế độ điều khiển Master / Slave
- Chuyển dữ liệu MSB First hoặc LSB First
- 7 bit rate có thể lập trình
- Cờ ngắt cuối chế độ truyền tải
- Write Collision Flag Protection
- Dánh thức khởi động chờ Idle
- Chế độ SPI Master (CK/2 ) Tốc độ kép

**Figure 75.** SPI Block Diagram



Note: Refer to Figure 1 on page 2 and Table 30 on page 74 for SPI pin placement.

Kết nối giữa các CPU Slave và master với SPI được chỉ ra trong hình 76 . Hệ thống thì bao gồm 2 thanh ghi Shift và bộ tạo xung nhịp Master . SPI Master khởi tạo 1

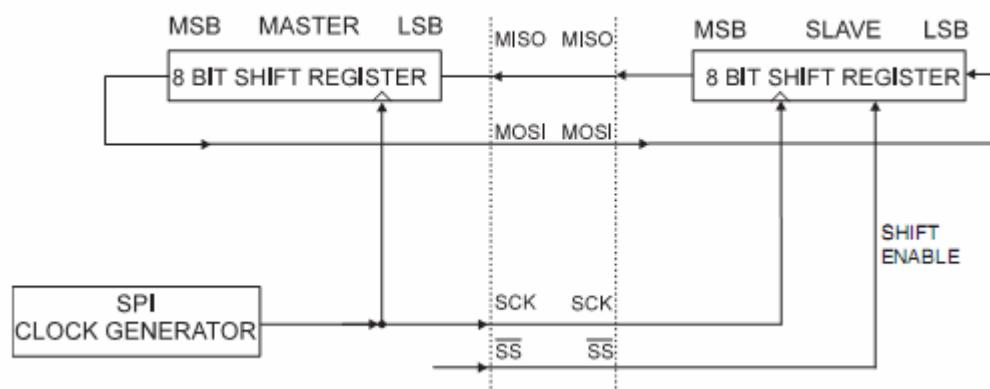
chu kì giao tiếp khi mà có sườn xung xuống của Slave lựa chọn chân SS của **Desired Slave**. Master và Slave chuẩn bị dữ liệu để gửi vào các thanh ghi Shift tương ứng của chúng, và master sinh ra 1 xung nhịp cần thiết trên nhánh SCK chuyển tiếp dữ liệu. Dữ liệu thì được chuyển từ Master sang Slave trên đầu ra Master Out -> đầu vào Slave In, MOSI, các nhánh, và từ Slave đến Master bằng chân Master In -> Slave Out, MISO, line. Sau mỗi gói dữ liệu, Master sẽ đồng bộ hóa với Slave bằng 1 xung cao lựa chọn Slave, Bít SS, Line.

Khi được cấu hình như là 1 Master, giao diện SPI không có sự tự động điều khiển của đường SS. Việc này phải được điều khiển bằng phần mềm người sử dụng trước khi quá trình giao tiếp có thể bắt đầu. Khi nó được thực hiện việc viết 1 Byte lên thanh ghi dữ liệu SPI sẽ khởi động bộ tạo xung nhịp SPI và phần cứng sẽ di chuyển 8 bít dữ liệu vào trong Slave. Sau khi di chuyển 1 Byte, bộ tạo xung nhịp SPI dừng lại, việc cài đặt cờ báo kết thúc quá trình chuyển dữ liệu (SPIF). Nếu như ngắt SPI kích hoạt bít (SPIE) trong thanh ghi SPCR được cài đặt, 1 ngắt được truy vấn. Master có thể tiếp tục chuyển byte dữ liệu tiếp theo bằng việc viết vào trong SPDR hoặc tín hiệu kết thúc của gói dữ liệu bằng xung cao được Slave lựa chọn, nhánh SS. Byte đến cuối cùng sẽ được giữ ở trong bộ đếm của thanh ghi cho lần sử dụng cuối cùng.

Khi được cấu hình như là Slave, giao diện SPI sẽ **remain sleeping** với 3 trạng thái MISO chỉ cần chân SS được điều khiển ở mức cao. Trong trạng thái này, phần mềm có thể cập nhật các thanh ghi dữ liệu SPI – SPDR, nhưng dữ liệu sẽ không được di chuyển ra ngoài bằng những xung nhịp đến trên chân SCK cho đến khi chân SS được điều khiển ở mức thấp. 1 byte vừa hoàn thành sự di chuyển thì cờ báo kết thúc sự di chuyển SPIF được cài đặt.

Nếu như ngắt SPI kích hoạt bít SPIE trong thanh ghi SPCR được cài đặt, 1 ngắt được yêu cầu. Slave có thể tiếp tục đặt dữ liệu mới để gửi vào trong thanh ghi SPDR trước khi quá trình đọc dữ liệu đến tiếp tục. Byte đến cuối cùng sẽ được dữ trong bộ đếm của thanh ghi sau lần sử dụng.

**Figure 76. SPI Master-Slave Interconnection**



Hệ thống được ghi vào bộ đếm đơn trong quá trình định hướng chuyển dữ liệu và được ghi vào bộ đếm kép trong quá trình định hướng dữ liệu đến. Điều này có nghĩa là các byte được di chuyển không thể được ghi vào thanh ghi dữ liệu SPI trước khi toàn bộ chu kì chuyển dời được hoàn thành. Khi dữ liệu đang đến, tuy nhiên 1 kí

tự đã đến phải được đọc từ thanh ghi dữ liệu SPI trước khi kí tự tiếp theo được di chuyển vào trong hoàn toàn . Nói cách khác bít đầu tiên bị mất .

Trong chế độ SPI Slave quá trình điều khiển logic sẽ lấy mẫu tín hiệu đến của chân SCK để đảm bảo việc lấy mẫu chính xác của tín hiệu xung nhịp mức thấp cực tiêu và mức cao của chu kì nên là :

- Chu kì thấp : ngắn hơn 2 lần chu kì xung nhịp CPU
- Chu kì cao : dài hơn 2 lần chu kì xung nhịp CPU

Khi mà SPI được kích hoạt , hướng dữ liệu của các chân MOSI , MISO ,SCK , SS được ghi đè theo bảng 69 . để biết thêm chi tiết về cổng tự động ghi đè tham khảo phần Alternate Port Function trên trang 71

**Table 69. SPI Pin Overrides<sup>(1)</sup>**

Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
SS	User Defined	Input

Đoạn code mẫu dưới đây chỉ ra cách để khởi tạo 1 SPI như là 1 Master và cách để tiến hành 1 quá trình chuyển dữ liệu đơn giản . DDR\_SPI trong các ví dụ phải được thay đổi bằng các chân SPI điều khiển thanh ghi định hướng dữ liệu thật sự . DD\_MOSI , DD\_MISO và DD\_SCK phải được thay thế bằng các bít định hướng dữ liệu thật sự cho các chân này . Ví dụ : nếu như MOSI được đặt lên chân PB5 , thay thế DD\_MOSI với DDB5 và DDR\_SPI với DDRB

Assembly Code Example <sup>(1)</sup>
<pre> SPI_MasterInit:     ; Set MOSI and SCK output, all others input     ldi r17,(1&lt;&lt;DD_MOSI) (1&lt;&lt;DD_SCK)     out DDR_SPI,r17     ; Enable SPI, Master, set clock rate fck/16     ldi r17,(1&lt;&lt;SPEN) (1&lt;&lt;MSTR) (1&lt;&lt;SPR0)     out SPCR,r17     ret  SPI_MasterTransmit:     ; Start transmission of data (r16)     out SPDR,r16     Wait_Transmit:     ; Wait for transmission complete     sbis SPSR,SPEIF     rjmp Wait_Transmit     ret </pre>
C Code Example <sup>(1)</sup>
<pre> void SPI_MasterInit(void) {     /* Set MOSI and SCK output, all others input */     DDR_SPI = (1&lt;&lt;DD_MOSI) (1&lt;&lt;DD_SCK);     /* Enable SPI, Master, set clock rate fck/16 */     SPCR = (1&lt;&lt;SPEN) (1&lt;&lt;MSTR) (1&lt;&lt;SPR0); }  void SPI_MasterTransmit(char cData) {     /* Start transmission */     SPDR = cData;     /* Wait for transmission complete */     while(!(SPSR &amp; (1&lt;&lt;SPEIF)))         ; } </pre>

Note: 1. See "About Code Examples" on page 9.

Đoạn code mẫu dưới đây chỉ ra cách để khởi tạo SPI như là 1 Slave và cách để tiến hành 1 quá trình tiếp nhận đơn giản

Assembly Code Example <sup>[1]</sup>
<pre> SPI_SlaveInit:     ; Set MISO output, all others input     ldi r17,(1&lt;&lt;DD_MISO)     out DDR_SPI,r17     ; Enable SPI     ldi r17,(1&lt;&lt;SPE)     out SPCR,r17     ret  SPI_SlaveReceive:     ; Wait for reception complete     sbis SPSR,SPIF     rjmp SPI_SlaveReceive     ; Read received data and return     in r16,SPDR     ret </pre>
C Code Example <sup>[1]</sup>
<pre> void SPI_SlaveInit(void) {     /* Set MISO output, all others input */     DDR_SPI = (1&lt;&lt;DD_MISO);     /* Enable SPI */     SPCR = (1&lt;&lt;SPE); }  char SPI_SlaveReceive(void) {     /* Wait for reception complete */     while(!(SPSR &amp; (1&lt;&lt;SPIF)))     {         /* Return data register */         return SPDR;     } } </pre>

Note: 1. See "About Code Examples" on page 9.

## Chức năng của chân SS

### Chế độ Slave

Khi SPI được cấu hình như là 1 Slave , chân lựa chọn Slave (SS) thì luôn luôn là đầu vào . Khi SS được dặt ở mức thấp SPI được kích hoạt và MISO trở thành 1 đầu ra nếu được cấu hình bởi người sử dụng . Tất cả các chân khác đều là đầu vào . Khi SS được điều khiển ở mức cao , tất cả các đầu vào chấp nhận MISO cái mà có thể được người sử dụng cấu hình như là 1 đầu ra và SPI bị động , điều này nghĩa là nó sẽ không nhận tín hiệu đến . Chú ý rằng SPI sẽ được reset 1 lần khi chân SS ở mức cao .

Chân SS thì hữu dụng cho việc đồng bộ hóa các gói và các bít để giữ bít Slave của bộ đến đị bộ với bộ tạo xung nhịp Master . Khi mà chân SS ở mức cao , SPI Slave sẽ ngay lập tức được Reset để gửi và nhận các giá trị Logic , và thả bất cứ phần dữ liệu đến vào trong thanh ghi Shift .

## Chế độ Master

Khi SPI được cấu hình như là 1 Master (Bit MSTR trong thanh ghi STCR được cài đặt = 1 ) người sử dụng có thể xác định hướng của cổng SS

Nếu chân SS được cấu hình như là 1 đầu ra chân này là 1 đầu ra chung cái mà không gây ảnh hưởng đến hệ thống SPI . Thông thường chân này sẽ được điều khiển trong chế độ SPI Slave

Nếu SS được cấu hình như là 1 đầu vào , nó phải được giữ ở mức cao để đảm bảo cho quá trình điều khiển SPI Master . Nếu như SS được điều khiển ở mức thấp bằng các mạch ngoại vi khi SPI được cấu hình như là 1 Master với chân SS được xác định như là , hệ thống SPI sẽ biên dịch điều này như là 1 Master khác lựa chọn SPI như là 1 Slave và nó được khởi động để gửi dữ liệu vào trong nó . Để tránh việc tranh giành bus , hệ thống SPI tạo ra các hành động dưới đây

- Bit MSTR trong thanh ghi STCR bị xóa và hệ thống SPI trở thành 1 Slave . Như là kết quả của việc SPI trở thành Slave các chân MOSI và SCK trở thành 1 đầu vào
- Cờ SPIF trong thanh ghi SPSR được cài đặt , và nếu ngắt của SPI được kích hoạt , bít I trong SREG được cài đặt thì các chương trình con phục vụ ngắt được thực thi

Vì vậy , khi ngắt điều khiển quá trình chuyển dữ liệu SPI được sử dụng trong chế độ Master , và ở đó khả năng tồn tại của chân SS ở mức thấp , các ngắt nên luôn luôn được kiểm tra xem là bít MSTR vẫn được cài đặt hay không . Nếu bít MSTR vừa bị xóa bằng chân lựa chọn chế độ Slave (Slave Select ) , nó phải được cài đặt bằng người sử dụng để kích hoạt lại chế độ SPI Master .

## Thanh ghi điều khiển SPI\_SPCR

Bít 7\_SPIE : kích hoạt ngắt SPI

Bít này gây ra gây ra ngắt SPI để thực thi nếu bít SPIF trong thanh ghi SPSR được đặt và nếu các bít kích hoạt chung trong thanh ghi SREG được đặt

Bít 6\_SPE : kích hoạt SPI

Khi bít SPE được khi là 1 thì SPI được kích hoạt . Bít này phải được cài đặt để kích hoạt bắt cứ quá trình điều khiển SPI nào

Bít 5\_DODR : thứ tự dữ liệu

Khi bít DODR được ghi là 1 LSB (bít thấp nhất) của từ dữ liệu (data word ) được di chuyển đầu tiên

Khi bít DODR được khi là 0 MSB (bít cao nhất) của từ dữ liệu (data word ) được di chuyển đầu tiên

Bít 4\_MSTR : lựa chọn chế độ Master Slave

Bít này lựa chọn chế độ Master SPI khi nó được ghi là 1 và chế độ Slave SPI khi nó được ghi là 0 . Nếu chân SS được cấu hình như là 1 đầu vào và được điều khiển ở mức thấp trong khi bít MSTR được cài đặt thì MSTR sẽ bị xóa và bít SPIF trong thanh ghi SPSR sẽ được cài đặt . Người sử dụng sau đó sẽ phải cài đặt bít MSTR để kích hoạt lại chế độ SPI Master

Bít 3\_CPOL : Clock polarity .

Khi bít này được viết là 1 , SCK ở mức cao khi trong chế độ Idle . Khi CPOL được viết là 0 thì SCK ở mức thấp khi bận . Tham khảo hình 77 . 78 như là 1 ví dụ . Chức năng của CPOL được liệt kê chi tiết bên dưới

**Table 70.** CPOL functionality

CPOL	Leading edge	Trailing edge
0	Rising	Falling
1	Falling	Rising

Bít 2\_CPHA : Pha xung nhịp

Việc cài đặt bít CPHA xác định nếu dữ liệu được lấy mẫu trên sườn đầu tiên hoặc sườn cuối cùng của SCK .Tham khảo hình 77 , 78 . Chức năng của CPHA được liệt kê bên dưới

**Table 71.** CPHA functionality

CPHA	Leading edge	Trailing edge
0	Sample	Setup
1	Setup	Sample

Bít 1,0\_SPR1,SPR0 : lựa chọn tốc độ xung nhịp SPI 1 hoặc 0

Hai bít này điều khiển tốc độ SCK của 1 thiết bị được cấu hình như là 1 Master . SPR1 và SPR0 không có hiệu lực ở trong chế độ Slave . Sự liên quan giữa SCK và tần số xung nhịp của bộ tạo giao động  $f_{OCS}$  được chỉ ra trong bảng dưới đây

**Table 72.** Relationship Between SCK and the Oscillator Frequency

SPI2X	SPR1	SPR0	SCK Frequency
0	0	0	$f_{osc}/4$
0	0	1	$f_{osc}/16$
0	1	0	$f_{osc}/64$
0	1	1	$f_{osc}/128$
1	0	0	$f_{osc}/2$
1	0	1	$f_{osc}/8$
1	1	0	$f_{osc}/32$
1	1	1	$f_{osc}/64$

## **Thanh ghi trạng thái SPI**

### Bít 7\_SPIF : Cờ ngắt SPI

Khi 1 sự chuyển dịch nối tiếp được hoàn thành, cờ SPIF được cài đặt . Một ngắt được sinh ra nếu SPIE trong thanh ghi SPCR được cài đặt và các ngắt chung được kích hoạt . Nếu như SS là 1 đầu vào và được điều khiển ở mức thấp khi mà SPI ở trong chế độ Master , điều này cũng sẽ cài đặt cờ SPIF . SPIF bị xóa bằng phần cứng khi mà đang thực thi các Vector điều khiển ngắt tương ứng . Như 1 sự lựa chọn , bít SPIF bị xóa bằng quá trình đọc đầu tiên của thanh ghi trạng thái SPI với SPIF được cài đặt , sau khi việc truy nhập vào thanh ghi SPI (SPDR )

### Bít 6\_WCOL : Viết cờ Collision

Bít WCOL được cài đặt nếu thanh ghi dữ liệu SPI (SPDR) được viết trong suốt quá trình chuyển đổi dữ liệu . Bít WCOL (và bít SPIF ) bị xóa bởi quá trình đọc đầu tiên của thanh ghi trạng thái SPI với WCOL được cài đặt và sau đó truy nhập vào thanh ghi trạng thái SPI .

### Bít 5...1\_RES : các bít dự trữ

Các bít này là các bít dự trữ ở trong Atmega128 và sẽ luôn luôn được đọc là 0

### Bít 0\_SPI2X : Bít tốc độ SPI kép

Khi bít này được ghi là 1 , tốc độ của SPI (tần số SCK) sẽ được gộp lại khi mà SPI ở trong chế độ Master (xem bảng 72) . Điều này có nghĩa là chu kỳ SCK cực tiểu sẽ bằng 2 lần chu kỳ xung nhịp của CPU . Khi SPI được cấu hình như là Slave , SPI chỉ đảm bảo để làm việc tại tần số  $f_{OSC}/4$  hoặc thấp hơn

Giao diện SPI ở trong Atmega 128 cũng được sử dụng cho bộ nhớ chương trình và việc download và upload dữ liệu của EEPROM . Xem bảng 300 để thêm chi tiết về phần lập trình nối tiếp SPI và sự xác minh (Verification )

## **Thanh ghi dữ liệu SPI\_SPDR**

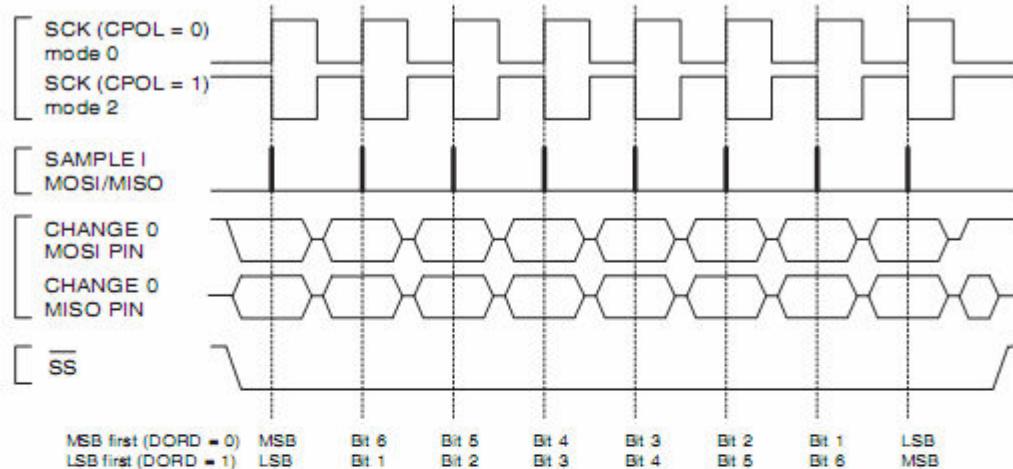
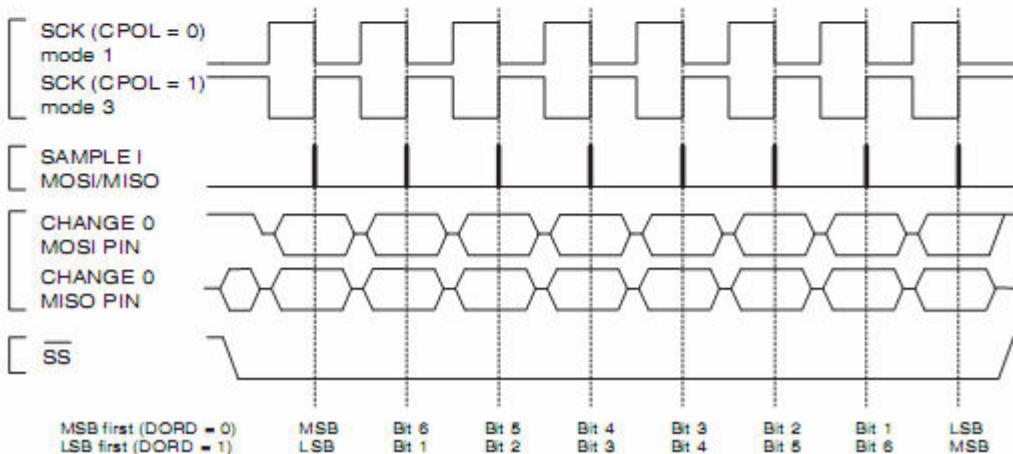
Thanh ghi dữ liệu SPI là 1 thanh ghi Read Write được sử dụng cho việc chuyển dữ liệu giữa các file đăng ký và các thanh ghi Shift SPI việc ghi lên các thanh ghi sẽ bắt đầu 1 quá trình chuyển dữ liệu . Việc đọc các thanh ghi gây lên việc bộ đệm dữ liệu đếm trong thanh ghi Shift được đọc

## **Các chế độ dữ liệu**

Có 4 sự kết hợp của pha SCK và polarity với sự tôn trọng đến các dữ liệu nối tiếp, cái mà được xác định bằng cách bít điều khiển CPHA và CPOL . Định dạng chuyển dữ liệu SPI thì được chỉ ra trong hình 77 , 78 . Các bít dữ liệu được di chuyển ra ngoài và được chốt ở sườn đối diện của tín hiệu SCK để đảm bảo đủ thời gian cho các tín hiệu trở về trạng thái ổn định . Điều này rõ ràng được nhìn thấy ở bảng 70 . 71

**Table 73.** CPOL and CPHA Functionality

	Leading edge	Trailing edge	SPI mode
CPOL = 0, CPHA = 0	Sample (Rising)	Setup (Falling)	0
CPOL = 0, CPHA = 1	Setup (Rising)	Sample (Falling)	1
CPOL = 1, CPHA = 0	Sample (Falling)	Setup (Rising)	2
CPOL = 1, CPHA = 1	Setup (Falling)	Sample (Rising)	3

**Figure 77.** SPI Transfer Format with CPHA = 0**Figure 78.** SPI Transfer Format with CPHA = 1

## XVII . USART

Bộ chuyển phát và thu nhận nối tiếp đồng bộ dị bộ vạn năng ( Universal Synchronous and Asynchronous serial Receiver and Transmitter – USART là 1 thiết bị truyền thông nối tiếp có độ linh hoạt cao . Các đặc điểm chính là :

- Hoạt động song công (full duplex) ( phụ thuộc vào các thanh ghi chuyển phát và thu nhận nối tiếp )
- Hoạt động đồng bộ hoặc dị bộ
- Hoạt động đồng bộ hóa khóa Master hoặc Slave
- Máy phát tốc độ Baud độ chính xác cao
- Hỗ trợ truyền các khung nối tiếp với 5 ,6 ,7, 8 hoặc 9 bit dữ liệu và 1 hoặc 2 bit stop
- Sự tạo bậc chẵn hoặc lẻ và hỗ trợ kiểm tra tính chẵn lẻ bằng phần cứng .
- Sự dò tràn dữ liệu
- Dò lỗi khung truyền
- Bộ lọc dài thấp kĩ thuật số và sự dò bit khởi động lỗi bao gồm bộ lọc nhiễu .
- 3 ngắt riêng biệt trên trọn bộ TX , trống thanh ghi dữ liệu TX , trọn bộ RX
- Chế độ truyền thông nhiều bộ sử lý
- Chế độ truyền thông dị bộ tốc độ kép

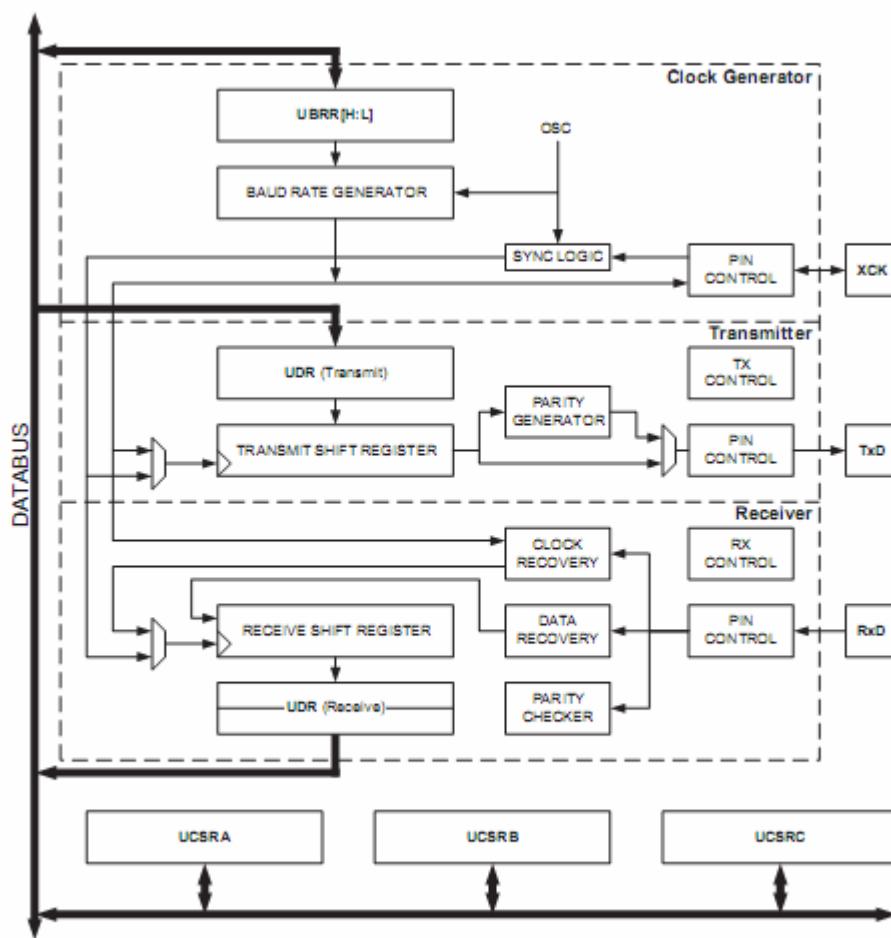
### USART kép – Dual USART

Atmega 128 có 2 bộ USART , là USART0 và USART1 . Chức năng của cả hai bộ USART thì được miêu tả bên dưới . USART0 và USART1 có các thanh ghi I/O khác nhau được chỉ ra trong phần “Register Summary “ ở trang 362 . Chú ý rằng trong chế độ tương thích với Atmega 103 , USART1 thì không khả dụng , chỉ có các thanh ghi là UBRR0H và UCRS0C . Điều này có nghĩa là trong chế độ tương thích với Atmega 103 , Atmega 128 hỗ trợ các quá trình điều khiển dị bộ của chỉ USART0

### Tổng quan

Một sơ đồ khối đã rút gọn của bộ chuyển phát USART được chỉ ra trong hình 79 . CPU có thể truy nhập vào các thanh ghi và các chân I/O được in đậm

Figure 79. USART Block Diagram



Note: Refer to Figure 1 on page 2, Table 36 on page 78, and Table 39 on page 81 for USART pin placement.

Các hộp nét đứt trong sơ đồ khối phân chia thành 3 phần chính của USART (được liệt kê từ trên xuống) : bộ phát xung nhịp , bộ chuyển phát , bộ thu nhận tín hiệu . Các thanh ghi điều khiển thì bị chia sẻ bởi tất cả các bộ phận . Khối logic phát xung nhịp thì bao gồm khối logic đồng bộ các đầu vào xung nhịp bên ngoài được sử dụng bởi quá trình điều khiển Slave đồng bộ , và máy phát baud rate . Chân XCK (xung nhịp chuyển ) thì chỉ được sử dụng bởi chế độ truyền đồng bộ . Bộ chuyển phát bao gồm 1 bộ đếm ghi đơn , 1 thanh ghi Shift nối tiếp , máy phát tương tự (Parity) và khối điều khiển logic cho việc điều khiển các dạng khung nối tiếp khác nhau . Bộ đếm ghi cho phép 1 sự truyền liên tiếp của dữ liệu mà không có bất cứ 1 độ trễ nào giữa các khung . Bộ thu nhận (Receiver) là bộ phận phức tạp nhất của module USART vì xung nhịp của và các bộ phận phục hồi dữ liệu. Các bộ phận khôi phục (Recovery) được sử dụng cho việc thu nhận dữ liệu 1 cách dị bộ . Thêm vào đó để khôi phục các bộ phận này , bộ thu nhận bao gồm bộ kiểm tra parity , khối logic điều khiển , 1 thanh ghi Register và 1 bộ đếm thu nhận 2 cấp (UDR) . Bộ thu nhận tín hiệu hỗ trợ các dạng khung giống nhau như là bộ chuyển phát , và có thể dò lỗi khung , tràn dữ liệu và các lỗi parity .

## AVR USART và AVR UART – chế độ tương thích

USART thì tương thích đầy đủ với AVR UART bất chấp :

- Địa chỉ bit bên trong của các thanh ghi USART
- Sự phát Baud Rate
- Chức năng bộ đệm chuyển phát
- Quá trình thu nhận tín hiệu

Tuy nhiên , bộ đệm của bộ thu nhận có 2 sự cải tiến rằng sẽ ảnh hưởng đến sự tương thích trong một vài trường hợp đặc biệt :

- 1 thanh ghi bộ đệm thứ 2 vừa được thêm vào . 2 thanh ghi bộ đệm hoạt động như 1 bộ đệm FIFO vòng tròn . Vì vậy UDR chỉ phải đọc một lần cho mỗi dữ liệu đến . Quan trọng hơn là nếu đó là các cờ báo lỗi (FE và DOR) và bit dữ liệu thứ 9 (RXB8) được ghi vào bộ đệm với dữ liệu ở trong bộ đệm của bộ thu tín hiệu . Vì vậy các bit trạng thái phải luôn được đọc trước khi thanh ghi UDR được đọc . Nói cách khác trạng thái lỗi sẽ mất do trạng thái bộ đệm bị mất .
- 1 thanh ghi Shift của bộ thu có thể đóng vai trò như một mức đệm thứ 3 . Điều này được thực hiện bằng việc cho phép dữ liệu đến tới vị trí còn lại trong thanh ghi Shift nối tiếp (xem hình 79 ) Nếu các thanh ghi bộ đệm bị đầy , cho đến khi 1 bit khởi động được tìm ra . USART vì vậy các điều kiện chống tràn dữ liệu (DOR) bền vững hơn .

Các bit điều khiển dưới đây đã được thay đổi tên , nhưng có chức năng giống và cùng địa chỉ thanh ghi :

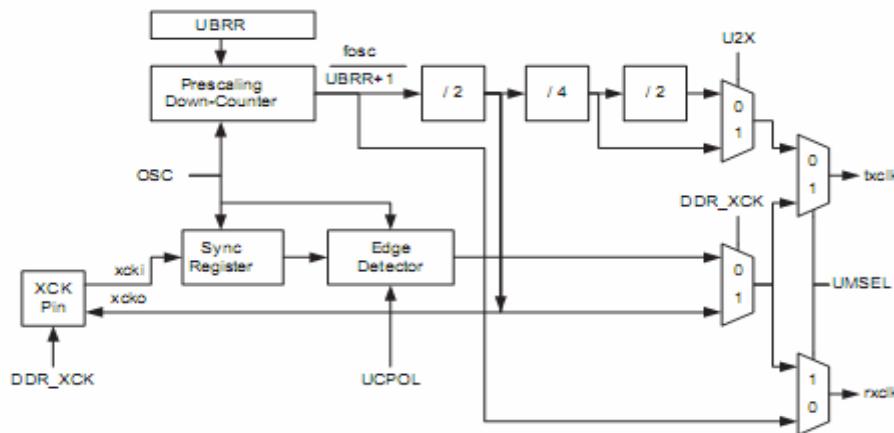
- CHR9 được đổi thành UCSZ2
- OR được đổi thành DOR

## Quá trình phát xung nhịp

Khối logic phát xung nhịp phát ra các xung nhịp cơ bản cho các bộ chuyển phát và bộ thu nhận tín hiệu . USART hỗ trợ 4 chế độ của quá trình điều khiển xung nhịp : dị bộ thông thường , dị bộ tốc độ kép , Master đồng bộ , và slave đồng bộ . Bit UMSEL trong thanh ghi C trạng thái và điều khiển USART (UCSRC) lựa chọn giữa quá trình điều khiển đồng bộ và dị bộ . Tốc độ kép (chỉ trong chế độ dị bộ ) thì được điều khiển dựa trên U2X trong thanh ghi UCSRA . Khi việc sử dụng chế độ đồng bộ (UMSEL=1) , thanh ghi định hướng dữ liệu cho chân XCK (DDR\_XCK) điều khiển lựa chọn nguồn xung nhịp là bên trong (Master Mode) hay bên ngoài (Slave Mode ) . Chân XCK này thì chỉ hoạt động khi sử dụng chế độ đồng bộ .

Figure 80 shows a block diagram of the clock generation logic.

**Figure 80. Clock Generation Logic, Block Diagram**



Sự miêu tả tín hiệu :

- txclk : xung nhịp bộ chuyển phát .(tín hiệu bên trong )
- rxclk : xung nhịp cơ bản của bộ thu nhận tín hiệu . (tín hiệu bên trong )
- xcki : đầu vào từ chân XCK (tín hiệu bên trong ) . Được sử dụng cho quá trình điều khiển slave đồng bộ
- xcko : xung nhịp đầu ra tới chân XCK (tín hiệu bên trong ) . Được sử dụng cho quá trình điều khiển đồng bộ
- - fosc : tần số chân XTAL ( xung nhịp hệ thống )

### Sự phát xung nhịp bên trong – máy phát Baud rate

Sự phát xung nhịp bên trong được sử dụng cho các chế độ master đồng bộ hóa và dị bộ của quá trình điều khiển . Sự miêu tả trong phần này tham khảo hình 80

Thanh ghi Baud rate USART (UBRR) và bộ đếm xuống được kết nối với tới cổng chức năng của nó như là một bộ đếm gộp trước có thể lập trình được hoặc máy phát baud rate . Bộ đếm xuống (down – counter), đang chạy ở tần số xung nhịp hệ thống (fosc) , được tải với giá trị của UBRR mỗi lần mà bộ đếm vừa đếm xuống đến 0 hoặc khi thanh ghi UBRRL được ghi . 1 xung nhịp được phát ra mỗi lần mà bộ đếm đạt đến giá trị 0 . Xung nhịp này là đầu ra máy phát baud rate ( $=f_{osc}/UBRR+1$ ) . Bộ chuyển phát chia đầu ra xung nhịp máy phát baud rate thành 2, 8 , 16 phụ thuộc vào chế độ . Đầu ra máy phát Baud rate được sử dụng 1 cách trực tiếp bởi xung nhịp của bộ thu nhận và các bộ phận khôi phục dữ liệu . Tuy nhiên , các bộ phận khôi phục dữ liệu sử dụng một state machine cái mà sử dụng 2 , 8, 16 trạng thái phụ thuộc vào chế độ được cài đặt bằng trạng thái của các bit UMSEL , U2X và DDR\_XCK

Bảng 74 bao gồm các công thức để tính toán baud rate (in bits per second ) và cho việc tính toán giá trị của UBRR cho mỗi chế độ của quá trình điều khiển của 1 nguồn xung nhịp được phát ra bên trong .

**Table 74. Equations for Calculating Baud Rate Register Setting**

Operating Mode	Equation for Calculating Baud Rate <sup>(1)</sup>	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{osc}}{16(UBRR + 1)}$	$UBRR = \frac{f_{osc}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{osc}}{8(UBRR + 1)}$	$UBRR = \frac{f_{osc}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{osc}}{2(UBRR + 1)}$	$UBRR = \frac{f_{osc}}{2BAUD} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps).

BAUD Baud rate (in bits per second, bps)

f<sub>osc</sub> System Oscillator clock frequency

UBRR Contents of the UBRRH and UBRL Registers, (0 - 4095)

Vài ví dụ của các giá trị UBRR cho vài tần số xung nhịp hệ thống được tìm ra trong bảng 82 (Xem bảng 194)

### Quá trình điều khiển tốc độ kép (U2X)

Tốc độ chuyển dữ liệu có thể được nhân đôi bằng việc cài đặt bit U2X trong thanh ghi UCSRA . Việc cài đặt bit này chỉ gây ảnh hưởng cho quá trình điều khiển dị bộ . Cài đặt bit này là 0 khi sử dụng chế độ điều khiển đồng bộ

Việc cài đặt bit này sẽ giảm ước chia của bộ chia baud rate từ 16 xuống 8 , nhân đôi 1 cách hiệu quả tốc độ truyền dữ liệu cho chế độ truyền thông không đồng bộ . Tuy nhiên chú ý rằng bộ thu tín hiệu trong trường hợp này sẽ chỉ sử dụng 1 nửa các lấy mẫu (giảm từ 16 xuống 8 ) cho việc lấy mẫu dữ liệu và khôi phục xung nhịp , vì vậy việc cài đặt baud rate chính xác hơn và xung nhịp hệ thống là cần thiết khi mà chế độ khi chế độ này được sử dụng . Về các bộ chuyển đổi , không có downside nào .

### Xung nhịp bên ngoài

Xung nhịp bên ngoài được sử dụng bởi quá trình điều khiển slave đồng bộ . Sự miêu tả trong phần này tham khảo hình 80 để biết thêm chi tiết

Dầu vào xung nhịp bên ngoài từ chân XCK thì được lấy mẫu bằng thanh ghi đồng bộ hóa để làm cực tiểu xác xuất của sự mất ổn định (meta-stability). Đầu ra từ thanh ghi đồng bộ hóa phải được đi qua 1 bộ dò sùn trước khi nó có thể được sử dụng bởi bộ chuyển phát và bộ thu nhận tín hiệu . Quá trình này đưa vào trong 2 trễ của chu kì xung nhịp CPU vì vậy tần số xung nhịp XCK bên ngoài cực đại thì được giới hạn bởi công thức sau

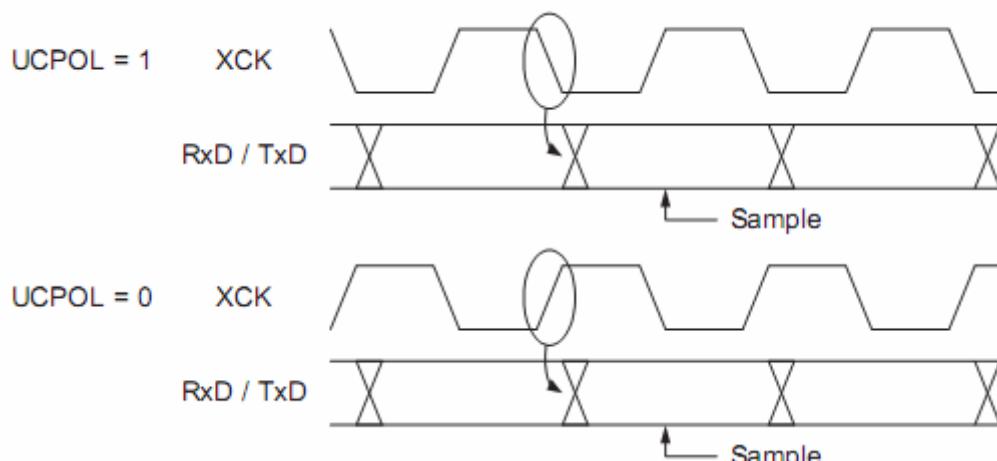
$$f_{XCK} < \frac{f_{osc}}{4}$$

Chú ý rằng  $f_{OSC}$  phụ thuộc vào độ ổn định của nguồn phát xung nhịp hệ thống . Vì vậy nó được khuyến cáo để thêm vào vài biên để tránh các sự mất mát có thể của sự biến thiên tần số dữ liệu .

### Điều khiển xung nhịp một cách đồng bộ

Khi chế độ đồng bộ được sử dụng ( $UMSEL = 1$ ) , chân XCK sẽ được sử dụng như là đầu vào xung nhịp (slave ) hoặc đầu ra xung nhịp (Master). Sự phụ thuộc giữa các sườn xung và sự lấy mẫu dữ liệu hoặc sự thay đổi dữ liệu là giống nhau . Nguyên tắc cơ bản là cái mà dữ liệu đầu vào (on RxD ) được lấy mẫu tại vị trí đối diện của sườn xung XCK của sườn đầu vào dữ liệu(TxD) bị thay đổi .

**Figure 81.** Synchronous Mode XCK Timing.



Các bit UCPOL và Bit UCRSC lựa chọn cái mà sườn xung XCK được sử dụng cho việc lấy mẫu dữ liệu và cái mà được sử dụng để thay đổi dữ liệu . Như hình 81 đã chỉ ra , khi UCPOL là 0 thì dữ liệu sẽ bị thay đổi tại sườn lên của xung XCK và được lấy mẫu tại sườn xuống của xung XCK . Nếu UCPOL được cài đặt , dữ liệu sẽ bị thay đổi tại sườn xuống của xung XCK và được lấy mẫu tại sườn lên của xung XCK

### Các dạng khung dữ liệu – Frame Formats

Một loạt các khung được xác định bằng 1 kí tự của các bit dữ liệu với các bit của bộ đồng bộ hóa (các bit start và stop ) , và một sự lựa chọn của các bit chẵn lẻ cho việc kiểm tra các lỗi . USART chấp nhận tất cả 30 sự kết hợp kế tiếp như là các dạng khung có hiệu lực

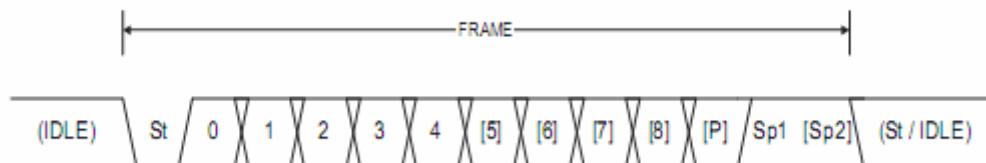
- 1 bit start
- 5 ,6 ,7 ,8 hoặc 9 bit dữ liệu
- Không có , hoặc có các bit chẵn lẻ
- 1 hoặc 2 bit stop

1 khung bắt đầu với bit start được theo sau là bit dữ liệu có nghĩa bé nhất . Sau đó tiếp theo là các bit dữ liệu , nâng tổng số lên 9 bit , và kế tiếp , kết thúc là bit có trọng số cao nhất . Nếu được kích hoạt , bit chẵn lẻ được chèn sau các bit dữ liệu , trước

các bit stop . Khi 1 khung hoàn thành được chuyển đi , nó có thể được dồn theo một khung mới , hoặc 1 đường giao tiếp nữa có thể được cài đặt trong trạng thái Idle (high )

Hình 82 minh họa sự kết hợp có thể của các dạng khung truyền . Các bit ở trong ngoặc là để lựa chọn .

**Figure 82. Frame Formats**



St bít start , luôn luôn ở mức thấp

(n) các bit dữ liệu (từ 0 đến 8 )

P bit chẵn lẻ . có thể là lẻ hoặc chẵn

Sp bit Stop , luôn ở mức cao

IDLE : không có quá trình chuyển phát nào trên đường giao tiếp dữ liệu (RxD hoặc TxD). Một đường IDLE phải ở mức cao

Dạng khung truyền được sử dụng bởi USART được cài đặt bằng bit UCSZ2:0, UPM1:0 và các bit USBS trong thanh ghi UCSRB và UCSRC . Bộ thu nhận và chuyển phát sử dụng các cài đặt giống nhau . Chú ý rằng việc thay đổi sự cài đặt của bất cứ bit nào trong số các bit này sẽ làm hỏng tất cả các giao tiếp đang tiến hành của cả bộ thu nhận và chuyển phát .

Các bit kích cỡ kí tự của USART lựa chọn số lượng của các bit dữ liệu trong một khung. Các bit Chế độ USART tương đương (UPM1:0) kích hoạt và cài đặt các loại của bit chẵn lẻ . Sự lựa chọn giữa 1 hoặc 2 bit Stop được thực hiện bằng bit lựa chọn bit Stop (USBS). Bộ thu nhận bỏ qua bit stop thứ hai . Một lỗi khung truyền (FE) vì vậy chỉ được tìm ra trong các khung nơi mà bit stop đầu tiên là 0

## Sự tính toán bit chẵn lẻ - Parity Bit Calculation

Bit chẵn lẻ thì được tính toán bằng việc exclusive-or của tất cả các bit dữ liệu . Nếu bit lẻ được sử dụng, kết quả của quá trình exclusive-or bị đảo ngược . Sự liên quan giữa bit chẵn lẻ và các bit dữ liệu như bên dưới :

$$\begin{aligned} P_{\text{even}} &= d_{n-1} \oplus \dots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0 \\ P_{\text{odd}} &= d_{n-1} \oplus \dots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1 \end{aligned}$$

$P_{\text{even}}$  bit chẵn lẻ sử dụng bậc chẵn

$P_{\text{odd}}$  bit chẵn lẻ sử dụng bậc lẻ

$d_n$  bit dữ liệu n của chuỗi kí tự

Nếu được sử dụng, bit chẵn lẻ được đặt giữa bit dữ liệu cuối và bit stop đầu tiên của chuỗi khung

## Sự khởi tạo USART

USART phải được khởi tạo trước khi bắt cứ một giao tiếp nào có thể xảy ra . Quá trình khởi tạo thông thường bao gồm việc cài đặt các baud rate , việc cài đặt các dạng khung và kích hoạt các bộ thu nhận và chuyển phát phụ thuộc vào yêu cầu sử dụng. Về hoạt động của các ngắt điều khiển USART , cờ báo ngắt chung nên bị xóa (và các ngắt chung bị vô hiệu hóa ) khi đang trong quá trình khởi tạo

Trước khi thực hiện 1 sự khởi tạo lại với các baud rate đã được thay đổi hoặc là các dạng khung truyền khác , phải đảm bảo rằng không có quá trình truyền phát nào đang tiến hành trong suốt giai đoạn mà thanh ghi bị thay đổi . Cờ TXC có thể được sử dụng để kiểm tra rằng bộ chuyển phát vừa hoàn thành tất cả các sự di chuyển , và cờ RXC có thể được sử dụng để kiểm tra rằng không có dữ liệu chưa được đọc trong bộ đệm của bộ thu nhận. Chú ý rằng cờ TXC phải được xóa trước mỗi quá trình truyền dữ liệu(trước UDR được ghi ) nếu nó được sử dụng cho chức năng này

Đoạn mã mẫu về sự khởi tạo USART đơn giản bên dưới chỉ ra một hàm C và assembly cái mà bằng nhau về chức năng . Các ví dụ thì giả định rằng chế độ điều khiển dị bộ sử dụng chế độ hỏi theo vòng(Polling) (không có ngắt nào được kích hoạt )và 1 dạng khung truyền ổn định . Baud rate thì được đưa ra như là 2 tham số chức năng. Về đoạn mã Assembly , tham số baud rate thì được giả định là được lưu trữ trong thanh ghi r17:r16

### Assembly Code Example<sup>(1)</sup>

```
USART_Init:
    ; Set baud rate
    out  UBRRH, r17
    out  UBRRL, r16
    ; Enable receiver and transmitter
    ldi  r16, (1<<RXEN) | (1<<TXEN)
    out  UCSRB,r16
    ; Set frame format: 8data, 2stop bit
    ldi  r16, (1<<USBS) | (3<<UCSZ0)
    out  UCSRC,r16
    ret
```

### C Code Example<sup>(1)</sup>

```
#define FOSC 1843200 // Clock Speed
#define BAUD 9600
#define MYUBRR FOSC/16/BAUD-1
void main( void )
{
    ...
    USART_Init ( MYUBRR );
    ...
}
void USART_Init( unsigned int ubrr )
{
    /* Set baud rate */
    UBRRH = (unsigned char)(ubrr>>8);
    UBRRL = (unsigned char)ubrr;
    /* Enable receiver and transmitter */
    UCSRB = (1<<RXEN)|(1<<TXEN);
    /* Set frame format: 8data, 2stop bit */
    UCSRC = (1<<USBS)|(3<<UCSZ0);
}
```

**Note:** 1. See "About Code Examples" on page 9.  
 More advanced initialization routines can be made that include frame format as parameters, disable interrupts and so on. However, many applications use a fixed setting of the baud and control registers, and for these types of applications the initialization code can be placed directly in the main routine, or be combined with initialization code for other I/O modules.

## Sự truyền dữ liệu – bộ chuyển phát USART

Một bộ chuyển phát USART được kích hoạt bằng việc cài đặt bit kích hoạt di chuyển (TXEN) trong thanh ghi UCSRB . Khi bộ chuyển phát được kích hoạt , chế độ điều khiển cổng bình thường của chân TxD bị ghi đè bằng USART và đưa ra các chức năng như là các cổng ra nối tiếp của bộ chuyển phát . Baud rate , chế độ điều khiển và các dạng khung truyền phải được cài đặt lên một lần trước khi thực hiện quá trình truyền dữ liệu . Nếu chế độ điều khiển đồng bộ được sử dụng , xung nhịp trên chân XCK sẽ bị ghi đè và được sử dụng như là xung nhịp của quá trình truyền dữ liệu

### Việc gửi các khung với 5 đến 8 bit dữ liệu

Một quá trình truyền dữ liệu được khởi tạo bằng việc tải bộ đệm phát với dữ liệu được phát . CPU có thể tải bộ đệm dữ liệu bằng việc viết lên vùng I/O UDR . Dữ liệu được ghi vào bộ đệm trong bộ đệm phát sẽ bị di chuyển vào thanh ghi Shift khi mà thanh ghi Shift sẵn sàng để gửi 1 khung mới . Thanh ghi Shift được tải với các dữ liệu mới nếu nó ở trong trạng thái Idle(không có sự chuyển phát dữ liệu nào đang tiến hành ) hoặc ngay lập tức sau khi bit stop cuối cùng của khung trước được phát đi . Khi thanh ghi Shift được tải với dữ liệu mới, nó sẽ phát 1 khung truyền hoàn chỉnh tại tốc độ được đưa ra bằng thanh ghi baud , bit U2X hoặc bằng XCK phụ thuộc vào chế độ điều khiển được sử dụng

Các đoạn code mẫu dưới đây chỉ ra một hàm truyền USART cơ bản dựa trên sự hỏi vòng của cờ báo trống thanh ghi dữ liệu (UDRE). Khi sử dụng các khung với ít hơn 8 ký tự , các bit có trọng số cao nhất được viết lên UDR được bỏ qua . USART phải được khởi tạo trước khi chức năng được sử dụng . Về đoạn mã Assembly , dữ liệu để gửi thì được coi như được lưu vào trong thanh ghi R16

**Assembly Code Example<sup>(1)</sup>**

```

USART_Transmit:
    ; Wait for empty transmit buffer
    sbis UCSRA,UDRE
    rjmp USART_Transmit
    ; Put data (r16) into buffer, sends the data
    out UDR,r16
    ret

```

**C Code Example<sup>(1)</sup>**

```

void USART_Transmit( unsigned char data )
{
    /* Wait for empty transmit buffer */
    while ( !( UCSRA & (1<<UDRE) ) )

    ;
    /* Put data into buffer, sends the data */
    UDR = data;
}

```

Note: 1. See "About Code Examples" on page 9..

Chức năng đơn giản đợi bộ đệm phát trả nên trống bằng việc kiểm tra cờ UDRE , trước khi tải nó với dữ liệu mới được phát đi . Nếu các ngắt báo trống thanh ghi dữ liệu được sử dụng , các chương trình con phục vụ ngắt sẽ viết dữ liệu lên bộ đệm

### **Việc gửi các khung với 9 bit dữ liệu**

Nếu chuỗi kí tự 9 bit được sử dụng (UCSZ = 7), bit thứ 9 phải được ghi lên bit TXB8 trong UCSRB trước khi byte thấp của chuỗi được ghi lên UDR . Đoạn code mẫu dưới đây chỉ ra một chức năng chuyển phát cái mà điều khiển chuỗi kí tự 9bit . Về code Assembly , dữ liệu được gửi thì được giả định là được lưu trữ trong các thanh ghi R17:R16

**Assembly Code Example<sup>(1)</sup>**

```

USART_Transmit:
    ; Wait for empty transmit buffer
    sbis UCSRA,UDRE
    rjmp USART_Transmit
    ; Copy 9th bit from r17 to TXB8
    cbi UCSRB,TXB8
    sbrc r17,0
    sbi UCSRB,TXB8
    ; Put LSB data (r16) into buffer, sends the data
    out UDR,r16
    ret

```

**C Code Example**

```

void USART_Transmit( unsigned int data )
{
    /* Wait for empty transmit buffer */
    while ( !( UCSRA & (1<<UDRE)) )

    /* Copy 9th bit to TXB8 */
    UCSRB &= ~(1<<TXB8);
    if ( data & 0x0100 )
        UCSRB |= (1<<TXB8);
    /* Put data into buffer, sends the data */
    UDR = data;
}

```

- Note: 1. These transmit functions are written to be general functions. They can be optimized if the contents of the UCSRB is static. I.e., only the TXB8 bit of the UCSRB Register is used after initialization.  
For I/O registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

Bit thứ 9 có thể được sử dụng cho việc hiển thị 1 khung địa chỉ khi sử dụng chế độ giao tiếp nhiều vi xử lí hoặc cho các giao thức điều khiển khác như là quá trình đồng bộ hóa .

### **Cò báo và ngắt của bộ chuyển phát**

Các bộ chuyển phát USART có 2 cò để hiển thị trạng thái của nó : báo trống thanh ghi dữ liệu USART (UDRE) và hoàn thành quá trình chuyển phát (TXC) . Cả hai cò có thể được sử dụng cho việc sinh ra các ngắt .

Cò báo trống thanh ghi dữ liệu (UDRE0 hiển thị khi mà bộ đệm chuyển phát sẵn sàng để nhận dữ liệu mới .Bit này được cài đặt khi mà bộ đệm phát trống , và bị xóa khi mà bộ đệm chuyển phát bao gồm dữ liệu đã chuyển đi cái mà vừa bị di chuyển vào trong thanh ghi Shift . Để tương thích với các thiết bị trong tương lai , luôn viết bit này là 0 khi viết thanhg hi UCSRA

Khi cờ báo ngắt trống thanh ghi dữ liệu kích hoạt (UDRIE) trong UCSRB được ghi là 1 , ngắt trống thanh ghi dữ liệu USART sẽ được thực thi chỉ cần UDRE được cài đặt( cung cấp các ngắt chung được cài đặt ) UDRE bị xóa bằng việc viết UDR. Khi ngắt điều khiển chế độ chuyển phát được sử dụng, chương trình con phục vụ ngắt báo trống thanh ghi dữ liệu phải được viết giá trị mới lên UDR để xóa UDRE hoặc vô hiệu hóa ngắt báo trống thanh ghi dữ liệu , nói cách khác 1 ngắt mới sẽ xuất hiện mỗi lần mà các ngắt được kết thúc

Bit cờ báo hoàn thành chuyển phát (TXC) được đặt là 1 khi mà khung truyền trọn vẹn trong thanh ghi chuyển phát Shift vừa được di chuyển ra ngoài và không có ngắt hiện hành nào được đưa ra trong bộ đệm chuyển phát . bit Cờ TXC thì tự động được xóa khi mà ngắt hoàn thành chuyển phát được thực thi , hoặc nó có thể bị xóa bằng việc viết là 1 lên bit địa chỉ của nó . Cờ TXC thì có ích trong các giao diện truyền thông bán song công (giống như chuẩn RS485) , ở đó 1 ứng dụng chuyển phát phải được đăng nhập vào chế độ nhận và giải phóng bus truyền thông ngay lập tức sau khi hoàn thành quá trình chuyển phát .

Khi bit kích hoạt ngắt hoàn thành chuyển phát (TXCIE) trong thanh ghi UCSRB được cài đặt , ngắt hoàn thành chuyển phát USART sẽ được thực thi khi mà cờ TXC trở thành được cài đặt , (được cung cấp khi mà các ngắt chung đã kích hoạt ) Khi ngắt hoàn thành quá trình chuyển phát được sử dụng , các chương trình con điều khiển ngắt không phải xóa cờ TXC , điều này được thực hiện một cách tự động .

## **Parity Generator (Máy phát chẵn lẻ )**

Máy phát chẵn lẻ tính toán bit chẵn lẻ cho các khung truyền dữ liệu nối tiếp . Khi mà bit chẵn lẻ được kích hoạt (UPM1=1) , bộ logic điều khiển chuyển phát chèn các bit chẵn lẻ giữa các bit dữ liệu cuối và bit stop đầu tiên của các khung truyền cái mà được gửi .

## **Sự vô hiệu hóa chuyển phát**

Sự vô hiệu hóa của quá trình chuyển phát (cài đặt TXEN là 0 ) sẽ không có hiệu lực cho đến khi quá trình chuyển phát đang chạy và đang chờ được hoàn thành . ví dụ : khi mà thanh ghi shift chuyển phát và thanh ghi bộ đệm chuyển phát không bao gồm dữ liệu được chuyển phát . Khi đã vô hiệu hóa , bộ chuyển phát sẽ không ghi đè lên chân TxD

## **Sự thu nhận dữ liệu – bộ thu USART**

Bộ thu USART được kích hoạt bằng cách viết lên bit kích hoạt bộ thu (RXEN) trong thanh ghi UCSRB là 1 . Khi bộ thu được kích hoạt , quá trình điều khiển chân thông thường của chan RxD được ghi đè bởi USART và đưa ra các chức năng như chân đầu vào nối tiếp của bộ thu phát . Baud rate , chế độ điều khiển và dạng khung truyền

phải được cài đặt một lần trước khi bắt cứ sự thu nhận nối tiếp nào có thể thực hiện . Nếu chế độ đồng bộ hóa được sử dụng , xung nhịp trên chân XCK sẽ được sử dụng như là xung chuyển phát .

### Sự thu nhận các khung truyền với 5 đến 8 bit dữ liệu

Bộ thu phát bắt đầu quá trình nhận dữ liệu khi mà nó dò thấy 1 bit khởi động có hiệu lực . Mỗi bit cái mà sau bit khởi động sẽ được lấy mẫu tại baud rate hoặc xung nhịp XCK , và được chuyển vào trong thanh ghi di chuyển nhận cho đến khi bit stop đầu tiên của khung được nhận . 1 bít stop thứ 2 sẽ bị bỏ qua bởi bộ thu nhận . Khi bit stop đầu tiên được nhận ví dụ : 1 khung nối tiếp hoàn chỉnh được đưa ra trong thanh ghi di chuyển nhận , thanh phần của thanh ghi shift sẽ bị di chuyển vào trong bộ đếm nhận . Bộ đếm nhận có thể được đọc sau đó bởi việc đọc vùng nhớ vào ra UDR

Đoạn code mẫu dưới đây chỉ ra như là một bộ thu nhận USART đơn giản chức năng được xây dựng trên cơ sở việc hỏi vòng (Polling) của cờ báo hoàn thành chuyển phát (RXC) . Khi sử dụng các khung truyền với ít hơn 8 bít , các bit có trọng số cao nhất của dữ liệu được đọc từ UDR sẽ được che lên 0 . USART được khởi tạo trước khi chức năng có thể được sử dụng

#### Assembly Code Example<sup>(1)</sup>

```
USART_Receive:
    ; Wait for data to be received
    sbis UCSRA, RXC
    rjmp USART_Receive
    ; Get and return received data from buffer
    in r16, UDR
    ret
```

#### C Code Example<sup>(1)</sup>

```
unsigned char USART_Receive( void )
{
    /* Wait for data to be received */
    while ( !(UCSRA & (1<<RXC)) )
        ;
    /* Get and return received data from buffer */
    return UDR;
}
```

Note: 1. See "About Code Examples" on page 9.

The function simply waits for data to be present in the receive buffer by checking the RXC flag, before reading the buffer and returning the value.

### Sự thu nhận các khung với 9 bit dữ liệu

Nếu chuỗi kí tự 9-bit được sử dụng (UCSZ = 7) bit thứ 9 phải được đọc tự bit RXB8 trong thanh ghi UCSRB trước khi việc đọc các bit thấp từ UDR . Nguyên tắc này

áp dụng lên các cờ báo trạng thái FE, DOR ,và UPE rất tốt . Đọc trạng thái từ UCSRA , sau đó dũ liệu từ UDR . Việc đọc vùng địa chỉ nhớ UDR sẽ thay đổi trạng thái của bộ đệm nhận FIFO và tiếp theo là các bit TXB8 , FE , DOR , UPE , tất cả được lưu trữ trong FIFO sẽ bị thay đổi .

Đoạn code mẫu dưới đây sẽ chỉ ra ví dụ đơn giản về chức năng thu nhận của USART cái mà điều khiển cả hai chuỗi kí tự 9 bit và các bit trạng thái .

Assembly Code Example <sup>[1]</sup>
<pre> USART_Receive:     ; Wait for data to be received     sbis UCSRA, RXC     rjmp USART_Receive     ; Get status and 9th bit, then data from buffer     in  r18, UCSRA     in  r17, UCSRB     in  r16, UDR     ; If error, return -1     andi r18, (1&lt;&lt;FE)   (1&lt;&lt;DOR)   (1&lt;&lt;UPE)     breq USART_ReceiveNoError     ldi  r17, HIGH(-1)     ldi  r16, LOW(-1)     USART_ReceiveNoError:     ; Filter the 9th bit, then return     lsr  r17     andi r17, 0x01     ret </pre>

C Code Example <sup>[1]</sup>
<pre> unsigned int USART_Receive( void ) {     unsigned char status, resh, resl;     /* Wait for data to be received */     while ( !(UCSRA &amp; (1&lt;&lt;RXC)) )         ;     /* Get status and 9th bit, then data */     /* from buffer */     status = UCSRA;     resh = UCSRB;     resl = UDR;     /* If error, return -1 */     if ( status &amp; (1&lt;&lt;FE)   (1&lt;&lt;DOR)   (1&lt;&lt;UPE) )         return -1;     /* Filter the 9th bit, then return */     resh = (resh &gt;&gt; 1) &amp; 0x01;      return ((resh &lt;&lt; 8)   resl); } </pre>

## Ngắt và cờ báo hoàn thành nhận

Bộ nhận USART có một cờ cái mà hiển thị trạng thái nhận tín hiệu

Cờ báo hoàn thành việc nhận tín hiệu ( RXC) hiển thị nếu có dữ liệu không được đọc đưa ra trong bộ đệm nhận . Cờ này là 1 khi dữ liệu không được đọc được xuất ra trong bộ đệm nhận , và 0 khi bộ đệm nhận trống (ví dụ không chứa bất cứ dữ liệu không được đọc nào ). Nếu bộ thu nhận bị vô hiệu hóa (RXEN = 0 ) thì bộ đệm thu nhận sẽ bị xóa sạch và tiếp theo bit RXC sẽ trở thành 0

Khi mà bit kích hoạt ngắt hoàn thành nhận(RXCIE) trong thanh ghi UCSRB được cài đặt , ngắt hoàn thành nhận USART sẽ được thực thi chỉ cần cờ RXC được cài đặt (được cung cấp khi các ngắt chung được kích hoạt ) Khi ngắt điều khiển nhận dữ liệu được sử dụng , chương trình con hoàn thành nhận phải đọc dữ liệu đến từ UDR để xóa cờ RXC , nói cách khác 1 ngắt mới sẽ xuất hiện mỗi lần mà chương trình con phục vụ ngắt hoàn thành .

Cờ báo lỗi nhận tín hiệu

Bộ nhận USART có 3 cờ báo lỗi : lỗi khung truyền (FE) , báo tràn dữ liệu (DOR) và báo lỗi chẵn lẻ (UPE) . Tất cả có thể được truy cập bằng cách đọc thanh ghi UCSRA . Thông thường các cờ báo là cái mà chúng được đặt trong bộ đệm nhận cùng với khung truyền cho các trạng thái lỗi của chúng được hiển thị . Dù cho bộ đệm của các cờ báo lỗi , UCSRA phải được đọc trước khi bộ đệm nhận (UDR), tiếp theo việc đọc vùng địa chỉ I/O UDR thay đổi bộ đệm đọc vùng dữ liệu . Sự tương đương khác cho các cờ báo lỗi là cái mà chúng không thể bị thay đổi bằng phần mềm đang thực hiện 1 quá trình viết lên vùng địa chỉ của cờ báo . Tuy nhiên , tất cả các cờ phải được cài đặt là 0 khi mà UCSRA được viết cho các sự tương thích của phía sau của các sự cài đặt USART trong tương lai . Không có các cờ báo lỗi nào có thể sinh ra các ngắt

Cờ báo lỗi khung truyền (FE) hiển thị trạng thái của bit STOP đầu tiên của khung truyền có thể đọc kế tiếp được lưu trữ trong bộ đệm nhận . Cờ FE là 0 khi bit Stop đã không được đọc đúng (như là 1 ) , và cờ báo FE sẽ là 1 khi bit stop không đúng (0) . Cờ này có thể được sử dụng cho việc dò các điều kiện out-of-sync , dò các điều kiện break , và các giao thức điều khiển . Cờ FE thì không bị ảnh hưởng bởi việc cài đặt của bit USBS trong thanh ghi UCSRC do đó bộ thu nhận bỏ qua tất cả , ngoại trừ các bit đầu tiên và các bit Stop . Để tương thích với các thiết bị trong tương lai , luôn cài đặt bit này là 0 khi đọc UCSRA

Cờ báo tràn dữ liệu (DOR) hiển thị việc mất dữ liệu dù cho bộ đệm dữ liệu đủ điều kiện. Một cờ báo tràn dữ liệu xuất hiện khi bộ đệm thu đầy(2 chuỗi kí tự ) , nó là 1 chuỗi kí tự mới đang đợi trong thanh ghi Shift Receiver , và 1 bit start mới được tìm thấy . Nếu cờ DOR được cài đặt đã có một hoặc nhiều khung truyền nối tiếp mất giữa khung cuối cùng được đọc từ UDR , và khung tiếp theo đọc từ UDR . Để tương thích với các thiết bị trong tương lai , luôn viết bit này là 0 khi viết lên UCSRA . Cờ báo DOR bị xóa khi khung đã nhận đã được di chuyển thành công khỏi thanh ghi Shift lên bộ đệm nhận .

Cờ báo lỗi chẵn lẻ (UPE) hiển thị cái mà khung truyền tiếp theo trong bộ đệm nhận có một lỗi chẵn lẻ khi đã nhận . Nếu sự kiểm tra không được kích hoạt , bit UPE sẽ luôn được đọc là 0 . Để tương thích với các thiết bị tương lai , luôn cài đặt bit này là 0 khi viết lên UCSRA . Để biết thêm chi tiết xem “Parity Bit Calculation” trang 176 và Parity checker trang 184 .

### **Bộ kiểm tra lỗi chẵn lẻ**

Bộ kiểm tra lỗi chẵn lẻ được hoạt động khi bit chế độ chẵn lẻ USART cao (UPM1) được cài đặt . Loại của sự kiểm tra chẵn lẻ được tiến hành (lẻ hoặc chẵn) được lựa chọn bằng bit UPM0 . Khi đã kích hoạt , bộ kiểm tra bit chẵn lẻ tính toán chẵn lẻ của các bit dữ liệu trên các khung truyền đang đến và so sánh kết quả với bit chẵn lẻ từ khung truyền nối tiếp . Kết quả của sự kiểm tra được lưu trữ trong bộ đệm nhận cùng với dữ liệu đến và các bit stop .Cờ báo lỗi chẵn lẻ (UPE) sau đó có thể đọc bằng phần mềm để kiểm tra nếu khung truyền có 1 lỗi chẵn lẻ .

Bit UPE được cài đặt nếu chuỗi ký tự tiếp theo cái mà có thể được đọc từ bộ đệm nhận có một lỗi chẵn lẻ khi đã nhận và việc kiểm tra lỗi chẵn lẻ được kích hoạt tại điểm đó (UPM1 =1 ) . Bit này thì có hiệu lực cho đến khi bộ đệm nhận(UDR) được đọc .

### **Vô hiệu hóa bộ thu nhận**

Trái ngược với bộ chuyển phát , việc vô hiệu hóa của bộ thu nhận sẽ là tức thời . Dữ liệu thu nhận đang đến sẽ vì vậy mà bị mất . Khi đã vô hiệu hóa (ví dụ : it RXEN được đặt là 0 ) bộ thu nhận sẽ không ghi đè lên chức năng thông thường của cổng RxD nữa . Bộ đệm nhận FIFO sẽ bị xóa sạch khi bộ thu nhận bị vô hiệu hóa . Dữ liệu còn lại trong bộ đệm sẽ bị mất

### **Xóa bộ đệm thu nhận**

Bộ đệm thu nhận FIFO sẽ bị xóa khi mà bộ thu nhận bị vô hiệu hóa , ví dụ : bộ đệm sẽ bị làm trống hết dữ liệu của nó . Các dữ liệu chưa được đọc sẽ bị mất . Nếu bộ đệm phải bị xóa trong suốt quá trình điều khiển thông thường (normal ) , do ví dụ của một lỗi trạng thái , đọc vùng nhớ I/O UDR cho đến khi cờ RXC bị xóa . Đoạn mã mẫu dưới đây chỉ ra cách để xóa bộ đệm nhận .

**Assembly Code Example<sup>(1)</sup>**

```
USART_Flush:
    sbis UCSRA, RXC
    ret
    in r16, UDR
    rjmp USART_Flush
```

**C Code Example<sup>(1)</sup>**

```
void USART_Flush( void )
{
    unsigned char dummy;
    while ( UCSRA & (1<<RXC) ) dummy = UDR;
}
```

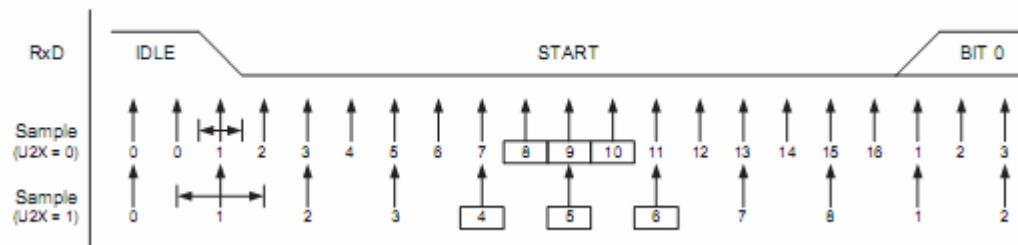
Note: 1. See "About Code Examples" on page 9..

The USART includes a clock recovery and a data recovery unit for handling asynchronous data reception. The clock recovery logic is used for synchronizing the internally generated baud rate clock to the incoming asynchronous serial frames at the RxD pin. The data recovery logic samples and low pass filters each incoming bit, thereby improving the noise immunity of the receiver. The asynchronous reception operational range depends on the accuracy of the internal baud rate clock, the rate of the incoming frames, and the frame size in number of bits.

## Sự khôi phục xung nhịp dị bộ

Khối logic khôi phục xung nhịp đồng bộ xung nhịp bên trong với các khung truyền nối tiếp đang đến. Hình 83 minh họa quá trình lấy mẫu của bit start của một khung truyền đang đến. Tốc độ lấy mẫu là 16 lần baud rate với chế độ normal , và 8 lần baud rate với chế độ tốc độ kép . Các mũi tên nằm ngang minh họa quá trình đồng bộ hóa độ lệch của quá trình lấy mẫu . Chú ý độ lệch thời gian lấy mẫu rộng hơn khi sử dụng chế độ tốc độ kép ( $U2X = 1$  ) của quá trình điều khiển . Các mẫu kí hiệu là 0 là các mẫu đã thực hiện khi mà đường RxD bị Idle (không có hoạt động truyền thông )

**Figure 83. Start Bit Sampling**

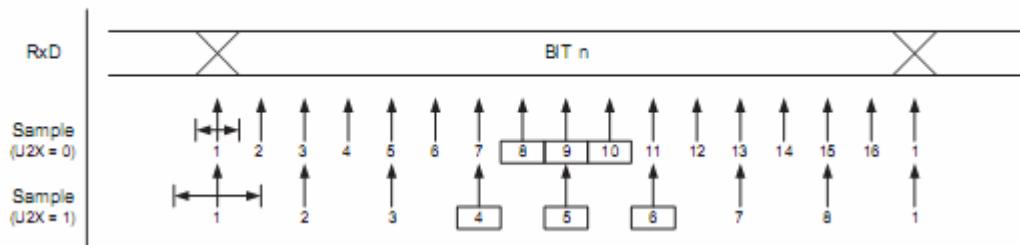


Khi khối logic khôi phục xung nhịp dò thấy 1 sự chuyển tiếp từ mức cao (idle ) sang mức thấp (start) trên đường RxD, quá trình dò bit start kế tiếp sẽ được khởi tạo . Kí hiệu lần lấy mẫu 1 là lần đầu tiên mẫu -0 như được chỉ ra trong hình . Khối logic khôi phục xung nhịp sau đó sẽ sử dụng các mẫu 8 ,9 ,10 cho chế độ normal , và các mẫu 4 ,5 ,6 cho chế độ tốc độ kép (đã hiển thị với số thứ tự các mẫu ở bên trong các hộp trên hình vẽ , để quyết định nếu 1 bit start có hiệu lực được nhận . Nếu nhiều hơn 2 hoặc 3 mẫu có các mức logic cao , bit start được giải phóng như là 1 định nghĩa và bộ thu nhận bắt đầu tìm 1 sự chuyển tiếp từ mức cao xuống mức thấp . Tuy nhiên , nếu 1 bit start có hiệu lực được dò thấy , khối logic khôi phục xung nhịp được đồng bộ hóa và quá trình khôi phục dữ liệu có thể bắt đầu . Quá trình đồng bộ hóa được lặp lại cho mỗi bit start

## Khôi phục dữ liệu dị bộ

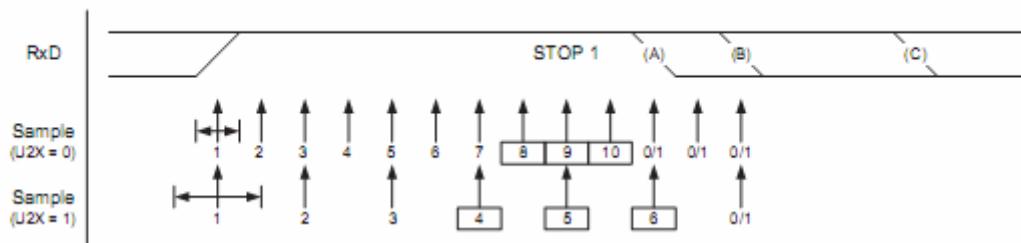
Khi xung nhịp bộ thu được đồng bộ hóa với bit start , sự khôi phục dữ liệu có thể bắt đầu . Bộ phận khôi phục dữ liệu có thể sử dụng một máy phát trạng thái ( state machine ) cái mà có 16 trạng thái cho mỗi bit trong chế độ bình thường và 8 trạng thái cho mỗi bit trong chế độ tốc độ kép . Hình 84 chỉ ra quá trình lấy mẫu của các bit dữ liệu và các bit chẵn lẻ . Mỗi mẫu thì đưa ra 1 số thứ tự cái mà bằng trạng thái của bộ phận khôi phục .

**Figure 84.** Sampling of Data and Parity Bit



Sự quyết định của các mức logic của bit đã thu nhận được đưa ra bằng việc thực hiện 1 quá trình bỏ phiếu đa số của giá trị logic lên 3 mẫu trong trung tâm của các bit đã nhận . Các mẫu trung tâm được làm nổi bật trong hình vẽ bằng việc cho các số thứ tự lấy mẫu vào trong các hộp. Quá trình bỏ phiếu lấy đa số được thực hiện như sau : Nếu 2 hoặc tất cả 3 mẫu có các mức logic cao , các bit đã nhận được đăng ký lên một mức logic 1 . Nếu 2 hoặc tất cả 3 mẫu có các mức logic thấp , các bit nhận được đăng ký lên một mức logic 0 . Quá trình bỏ phiếu lấy đa số này đóng vai trò như là 1 bộ lọc thông thấp cho các tín hiệu đến trên chân RxD . Quá trình khôi phục sau đó được lập lại cho đến khi một khung truyền hoàn chỉnh được nhận . Bao gồm cả bit stop đầu tiên .Chú ý rằng bộ nhận chỉ sử dụng bit stop đầu tiên của một khung truyền , Hình 85 chỉ ra quá trình lấy mẫu của bit stop và sự bắt đầu sớm nhất có thể của bit start của khung kế tiếp .

**Figure 85.** Stop Bit Sampling and Next Start Bit Sampling



Quá trình bỏ phiếu đa số cũng được thực hiện lên bit stop như đã thực hiện với các bit khác trong khung truyền . Nếu bit stop được đăng ký lại để có giá trị logic 0 ,cò báo lỗi khung truyền sẽ được cài đặt .

Một quá trình chuyển dịch từ cao xuống thấp mới đang hiển thị bit start của 1 khung truyền mới có thể đến sau khi bit cuối cùng được sử dụng cho quá trình bỏ phiếu đa số . Đối với chế độ tốc độ bình thường , mẫu mức thấp đầu tiên có thể được đánh dấu điểm A trong hình vẽ . Đối với chế độ tốc độ kép , mức thấp đầu tiên phải được làm trễ là B. (C) đánh dấu 1 bit stop đủ độ dài . Sự dò bit start sớm ảnh hưởng đến dải hoạt động của bộ thu nhận.

## Dải hoạt động dị bộ

Dải hoạt động dị bộ của bộ thu nhận thì phụ thuộc vào sự không khớp giữa tốc độ bit đã nhận và baudrate đã phát ra bên trong .Nếu bộ thu phát đang gửi các khung tại các tốc độ bit quá nhanh hoặc quá chậm , hoặc baud rate được phát ra bên trong của bộ thu nhận không có một sự tương đồng (xem bảng 75) về tần số cơ bản , Bộ thu nhận sẽ không thể đồng bộ hóa các khung với bit start

Công thức dưới đây có thể được sử dụng để tính toán hệ số của tốc độ dữ liệu đang đến và tốc độ bộ nhận bên trong :

$$R_{slow} = \frac{(D + 1)S}{S - 1 + D \cdot S + S_F}$$

$$R_{fast} = \frac{(D + 2)S}{(D + 1)S + S_M}$$

D Sum of character size and parity size (D = 5 to 10-bit)

S Samples per bit. S = 16 for Normal Speed mode and S = 8 for Double Speed mode.

$S_F$  First sample number used for majority voting.  $S_F = 8$  for Normal Speed and  $S_F = 4$  for Double Speed mode.

$S_M$  Middle sample number used for majority voting.  $S_M = 9$  for Normal Speed and  $S_M = 5$  for Double Speed mode.

$R_{slow}$  is the ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate.  $R_{fast}$  is the ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate.

Table 75 and Table 76 list the maximum receiver baud rate error that can be tolerated. Note that normal speed mode has higher toleration of baud rate variations.

**Table 75.** Recommended Maximum Receiver Baud Rate Error for Normal Speed Mode (U2X = 0)

D # (Data+Parity Bit)	R <sub>slow</sub> %	R <sub>fast</sub> %	Max Total Error %	Recommended Max Receiver Error %
5	93,20	106,67	+6.67/-6.8	± 3.0
6	94,12	105,79	+5.79/-5.88	± 2.5
7	94,81	105,11	+5.11/-5.19	± 2.0
8	95,36	104,58	+4.58/-4.54	± 2.0
9	95,81	104,14	+4.14/-4.19	± 1.5
10	96,17	103,78 %	+3.78/-3.83	± 1.5

**Table 76.** Recommended Maximum Receiver Baud Rate Error for Double Speed Mode (U2X = 1)

D # (Data+Parity Bit)	R <sub>slow</sub> %	R <sub>fast</sub> %	Max Total Error %	Recommended Max Receiver Error %
5	94,12	105,66	+5.66/-5.88	± 2.5
6	94,92	104,92	+4.92/-5.08	± 2.0
7	95,52	104,35	+4.35/-4.48	± 1.5
8	96,00	103,90	+3.90/-4.00	± 1.5
9	96,39	103,53	+3.53/-3.61	± 1.5
10	96,70	103,23	+3.23/-3.30	± 1.0

Sự khuyến cáo đối với lỗi tốc độ baud rate cực đại được đưa ra với giả thiết rằng bộ thu và bộ phát được chia bằng nhau tổng cực đại các lỗi

Có hai nguồn có thể gây ra lỗi baudrate các bộ thu nhận . Xung nhịp hệ thống của bộ thu nhận (XTAL) sẽ luôn có vài trạng thái nhỏ nhất trên giải điện áp nguồn cấp và giải nhiệt độ cho phép. Khi sử dụng một bộ tạo dao động thạch anh để phát ra xung nhịp hệ thống, điều này hiếm khi có vấn đề nhưng một bộ khuyêch đại dao động hệ thống có thể lớn hơn 2% phụ thuộc vào sai số cho phép của bộ khuyêch đại dao động. Nguyên nhân thứ 2 cho lỗi này là có thể điều khiển hơn. Máy phát dao động không thể luôn thực hiện một phép chia chính xác cho tần số hệ thống để có được tốc độ mong muốn. Trong trường hợp này ,một giá trị UBRR được đưa ra một giá trị có thể chấp nhận , lỗi thấp có thể được sử dụng nếu có thể

## Chế độ truyền thông đa vi xử lý

Việc cài đặt truyền thông đa vi xử lý (MPCM) Trong thanh ghi UCSRA kích hoạt một chức năng kích hoạt của các khung truyền đến đã nhận bởi bộ thu USART. Các khung này cái mà không bao gồm thông tin địa chỉ sẽ được bỏ qua và không được đặt vào trong bộ đệm thu . Hiệu quả của việc này làm giảm số lượng của các khung truyền đến cái mà phải được điều khiển bởi CPU , trong hệ thống với nhiều MCU mà giao tiếp thông qua các bit nối tiếp giống nhau . Bộ chuyển phát thì không bị ảnh hưởng

bởi việc cài đặt MPCM , nhưng phải được sử dụng khác nhau khi nó là 1 phần của hệ thống đang dùng chế độ truyền thông đa vi xử lý .

Nếu bộ thu nhận được cài đặt lên các khung truyền nhận mà bao gồm 5 đến 8 bit dữ liệu , sau đó bit stop đầu tiên hiển thị nếu khung truyền bao gồm dữ liệu hoặc thông tin địa chỉ . Nếu bộ thu nhận được cài đặt cho các khung với 9 bit dữ liệu , sau đó bit thứ 9 (TXB8) được sử dụng cho việc đặt địa chỉ riêng và các khung dữ liệu . Khi bit loại khung (bit stop đầu tiên hoặc bit dữ liệu thứ 9 ) là 1 , khung truyền bao gồm 1 địa chỉ . Khi bit loại khung là 0 thì khung là 1 khung dữ liệu .

Chế độ truyền thông đa vi xử lý kích hoạt hàng loạt MCU slave tới dữ liệu nhận từ một MCU master . Điều này được thực hiện bởi mã dịch đầu tiên 1 khung truyền địa chỉ để tìm ra cái mà MCU được đặt địa chỉ . Nếu 1 MCU slave đặc biệt vừa được đặt địa chỉ , nó sẽ nhận khung dữ liệu tiếp theo như thông thường , trong khi các MCU slave khác sẽ bỏ qua các khung truyền đã nhận cho đến khi khung truyền địa chỉ khác được nhận

## Sử dụng MPCM

Một MCU để đóng vai trò như là 1 MCU master , nó có thể sử dụng dạng khung chuỗi kí tự 9 bit (UCSZ =7). Bit thứ 9 (TXB8) phải được cài đặt khi một khung địa chỉ (TXB8=1) hoặc bị xóa khi 1 khung dữ liệu (TXb=0) đang được chuyển phát . Các MCU slave phải trong trường hợp phải được cài đặt lên dạng khung truyền chuỗi kí tự 9 bit

Quy trình sau đây nên được sử dụng để chuyển đổi dữ liệu tổng chế độ truyền thông đa vi xử lý :

- Tất cả các MCU slave đều trong chế độ truyền thông đa vi xử lý (MPCM) trong thanh ghi UCSRA được cài đặt 0
- MCU master gửi 1 khung địa chỉ , và tất cả các slave nhận và đọc khung truyền này . Trong các MCU slave , cờ RXC trong thanh ghi UCSRA sẽ được cài đặt như thông thường .
- Mỗi MCU slave đọc thanh ghi UDR và xác định nếu nó vừa được lựa chọn . Vì vậy nếu nó xóa bit MPCM trong thanh ghi UCSRA , nói cách khác , nó đợi cho byte địa chỉ kế tiếp và giữ việc cài đặt MPCM
- Việc đặt địa chỉ MCU sẽ nhận tất cả các khung truyền dữ liệu cho đến khi 1 khung truyền địa chỉ mới được nhận. Các MCU slave khác , cái mà vẫn có bit MPCM được cài đặt , sẽ bỏ qua các khung truyền dữ liệu .
- Khi mà các khung truyền dữ liệu được nhận bởi các MCU đã đặt địa chỉ , các MCU đặt địa chỉ cài đặt bit MPCM và đợi một khung truyền địa chỉ mới từ master . Quá trình này sau đó được lặp lại từ bước 2

Việc sử dụng bất cứ các dạng khung truyền chuỗi kí tự từ 5 đến 8 bit là có thể , nhưng không thực tế do đó các bộ thu phải được thay đổi giữa việc sử dụng các dạng khung truyền chuỗi kí tự n và n+1 . Điều này làm cho quá trình điều khiển song công khó khăn hơn do đó bộ thu nhận và chuyên phát sử dụng sự cài đặt các

cõ khung truyền giống nhau . Nếu các khung truyền chuỗi kí tự từ 5 đến 8 bit được sử dụng , bộ chuyển phát phải được cài đặt để sử dụng 2 bit stop (USBS = 1) do đó bit stop đầu tiên được sử dụng để hiển thị các dạng khung truyền giống nhau

Không sử dụng lệnh read-modify-write (SBI và CBI) để cài đặt hoặc xóa bit MPCM . Bit MPCM chia sẻ các vùng địa chỉ I/O như là cõ TXC và điều này có thể xảy ra biến cố bị xóa khi sử dụng lệnh CBI và SBI

## Sự miêu tả thanh ghi USART

### Thanh ghi dữ liệu vào ra USARTn

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	UDRn (Read) UDRn (Write)							
Initial Value	0	0	0	0	0	0	0	0	

Các thanh ghi bộ đệm dữ liệu phát USARTn và các thanh ghi bộ đệm dữ liệu thu nhận chia sẻ các địa chỉ vào ra giống nhau được tham chiếu như là thanh ghi bộ đệm dữ liệu USART hoặc UDRn . Thanh ghi bộ đệm dữ liệu chuyên phát (TXBn) sẽ định hướng đến cho dữ liệu được viết lên vùng nhớ thanh ghi UDRn . Việc đọc vùng nhớ thanh ghi UDRn sẽ hoàn trả các thành phần của thanh ghi bộ đệm dữ liệu thu nhận

Về các chuỗi kí tự 5 , 6 ,7 bit hoặc nhiều bit không được sử dụng sẽ bị bỏ qua bởi bộ chuyển phát và được đặt là 0 bằng bộ thu nhận .

Bộ đệm phát có thể chỉ được viết khi cõ báo UDREn trong thanh ghi UCSRA được đặt . Dữ liệu được viết lên thanh ghi UDRn khi mà cõ báo UDREn không được cài đặt , sẽ bị bỏ qua bởi bộ chuyển phát USARTn . Khi dữ liệu được viết lên bộ đệm phát , và bộ chuyển phát được kích hoạt , bộ chuyển phát sẽ tải dữ liệu vào troang thanh ghi Shift của bộ chuyển phát khi mà thanh ghi Shift còn trống . Sau đó dữ liệu sẽ được phát nối tiếp trên chân TxDn

Bộ đệm thu nhận thì bao gồm 1 FIFO 2 mức . FIFO sẽ thay đổi trạng thái của nó ở nơi mà bộ đệm thu nhận được truy cập đến . Dù cho quá trình xử lý của bộ đệm này , không sử dụng đọc thay cho các lệnh viết (SBI và CBI ) trong vùng nhớ này . Phải cẩn thận khi sử dụng các lệnh kiểm tra bit (SBIC và SBIS) , do đó các bit này cũng sẽ thay đổi trạng thái của FIFO

### Thanh ghi A trạng thái và điều khiển USART – UCSRnA

Bit	7	6	5	4	3	2	1	0	
Read/Write	R	R/W	R	R	R	R	R/W	R/W	UCSRnA
Initial Value	0	0	1	0	0	0	0	0	

Bit 7 – RCXn : hoàn thành thu nhận tín hiệu

Bit cờ này được cài đặt khi có các dữ liệu không được đọc trong bộ đệm nhận và bị xóa khi bộ đệm nhận trống (ví dụ không bao gồm bất cứ bit chưa được đọc nào) . Nếu bộ thu nhận bị vô hiệu hóa , bộ đệm thu sẽ bị xóa sạch và tiếp theo bit RXCn sẽ trở thành 0.

Cờ báo RXCn có thể được sử dụng để phát ra 1 ngắt báo hoàn thành nhận (xem miêu tả của bit RXCIEn )

#### Bit 6 – TXCn : hoàn thành chuyển dữ liệu USART

Bit cờ này được cài đặt để đảm bảo rằng khung trong thanh ghi Shift chuyển phát vừa được nén lại và không có dữ liệu mới hiện hành nào được đưa ra trong bộ đệm chuyển phát (UDRn) . Bít cờ báo TXCn thì tự động được xóa khi mà ngắt báo chuyển phát hoàn thành được thực thi hoặc nó có thể bị xóa bằng việc viết bit 1 mức logic một lên vùng bit nhớ của nó . Cờ báo TXCn có thể phát ra 1 ngắt báo hoàn thành chuyển phát (xem miêu tả của bit TXCIEn )

#### Bit 5 – UDREn : báo trống thanh ghi dữ liệu USART

Cờ báo UDREn hiển thị nếu bộ đệm phát (UDRn) sẵn sang để nhận dữ liệu mới . Nếu UDREn là một , bộ đệm là trống , và vì vậy sẵn sàng để được ghi . Cờ báo UDREn có thể sinh ra một ngắt báo trống thanh ghi dữ liệu(xem miêu tả của bit UDRIEn )

UDREn được cài đặt sau khi 1 reset hiển thị rằng bộ chuyển phát đã sẵn sàng

#### Bit 4 – Fen : lỗi khung truyền

Bit này được cài đặt nếu chuỗi kí tự tiếp theo trong bộ đệm nhận có lỗi khung truyền khi được nhận tín hiệu ví dụ như khi bit stop đầu tiên của chuỗi kí tự tiếp theo trong bộ đệm nhận là 0 . Bit này thì có hiệu lực cho đến khi bộ đệm nhận được đọc . Bit Fen là 0 khi bit stop của dữ liệu đã nhận là 1 . Luôn cài đặt bit này là 0 khi viết UCSRnA

#### Bit 3 – DORn : báo tràn dữ liệu

#### Bit 4 – Fen : lỗi khung truyền

Bit này được cài đặt nếu chuỗi kí tự tiếp theo trong bộ đệm nhận có lỗi khung truyền khi được nhận tín hiệu ví dụ như khi bit stop đầu tiên của chuỗi kí tự tiếp theo trong bộ đệm nhận là 0 . Bit này thì có hiệu lực cho đến khi bộ đệm nhận được đọc . Bit Fen là 0 khi bit stop của dữ liệu đã nhận là 1 . Luôn cài đặt bit này là 0 khi viết UCSRnA

#### Bit 3 – DORn : báo tràn dữ liệu

Bit này được cài đặt nếu một điều kiện tràn dữ liệu được tìm thấy . Một báo tràn dữ liệu xuất hiện khi bộ đệm thu bị đầy (2 chuỗi kí tự) , nó là 1 chuỗi kí tự mới đợi trong thanh ghi Shift thu nhận , và một bit khởi động mới được tìm thấy . Bit này có hiệu lực cho đến khi bộ đệm nhận (UDRn) được đọc . Luôn cài đặt bit này là 0 khi viết lên UCSRnA

#### Bit 2 – UPEn : lỗi chẵn lẻ

Bit này được cài đặt nếu chuỗi kí tự tiếp theo trong bộ đệm nhận có lỗi chẵn lẻ khi đã nhận và sự kiểm tra chẵn lẻ được kích hoạt tại điểm này (UPMn1=1) . Bit này có

hiệu lực cho đến khi bộ đệm nhận (UDRn) được đọc . Luôn cài bit này là 0 khi viết lên UCSRnA

Bit 1 – U2Xn: tốc độ truyền dữ liệu USART kép

Bit này chỉ có hiệu lực trong chế độ điều khiển dị bộ . Viết bit này là 0 khi sử dụng chế độ đồng bộ .

Viết bit này là 1 sẽ giảm hệ số chia của bộ chia baud rate từ 16 xuống 8 có tác dụng nhân đôi tốc độ truyền dữ liệu cho chế độ truyền thông dị bộ

Bit 0 – MPCMn : chế độ truyền thông đa vi xử lý

Bit này kích hoạt chế độ truyền thông đa vi xử lý . Khi bit MPCMn được viết là 1 , tất cả các khung truyền đến được nhận bằng bộ thu nhận USART cái mà không bao gồm thông tin địa chỉ sẽ bị bỏ qua . Bộ chuyển phát thì không bị ảnh hưởng bởi việc cài đặt MPCMn . Để biết thêm chi tiết xem (trang 187)

## Thanh ghi B trạng thái và điều khiển USARTn – USARTnB

Bit	7	6	5	4	3	2	1	0	UCSRnB
ReadWrite	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – RXCIE : kích hoạt ngắt hoàn thành RX

Việc viết bit này là 1 kích hoạt ngắt trên cờ RXC . 1 ngắt hoàn thành nhận USART sẽ chỉ được sinh ra nếu bit RXCIE được viết là 1 , các cờ báo ngắt chung trong thanh ghi SREG được viết là 1 và bit RXC trong thanh ghi UCSRnA được cài đặt .

Bit 6 – TXCIE : kích hoạt ngắt hoàn thành TX

Việc viết bit này kích hoạt ngắt trên cờ TXCn . 1 ngắt hoàn thành quá trình chuyển USART sẽ chỉ được sinh ra nếu bit TXCIE được viết là 1 , các cờ báo ngắt chung trong thanh ghi SREG thì được viết lên 1 và bit TXCn trong thanh ghi UCSRnA được cài đặt .

Bit 5 – UDRIEn : kích hoạt ngắt trống thanh ghi dữ liệu USART

Việc viết bit này lên 1 kích hoạt các ngắt trên cờ UDREn . 1 ngắt trống thanh ghi dữ liệu sẽ chỉ được sinh ra nếu bit UDRIEn được viết lên 1 , cờ ngắt chung trong thanh ghi SREG được viết lên 1 và bit UDREn trong UCSRnA được cài đặt

Bit 4 – RXENn : kích hoạt bộ thu tín hiệu

Việc viết các bit này lên 1 kích hoạt bộ thu USARTn . Bộ thu sẽ ghi đè lên quá trình điều khiển cổng thông thường cho chân RxDn khi được kích hoạt . Việc vô hiệu hóa bộ thu sẽ xóa sạch bộ đệm thu nhận không có hiệu lực Fen , và các cờ DORn và UPEn

Bit 3 – TXENn : kích hoạt bộ chuyển phát

Việc viết bit này là 1 kích hoạt bộ chuyển phát USARTn . Bộ chuyển phát sẽ ghi đè lên cổng điều khiển thông thường cho chân TxDn khi đã kích hoạt . Sự vô hiệu hóa của bộ chuyển phát (viết TXENn là 0 ) sẽ không trở nên có hiệu lực cho đến khi đang được tiến hành và quá trình truyền dữ liệu được hoàn thành . ví dụ như Khi thanh ghi

Shift chuyển phát thanh ghi bộ đệm phát không chứa dữ liệu được chuyển . Khi đã vô hiệu hóa , bộ chuyển phát sẽ ghi đè lâu hơn lên cổng TxDrn

Bit 2 – UCSZn2 : kích cỡ chuỗi kí tự

Các bit UCSZn2 được kết hợp với bit UCSZn1:0 trong thanh ghi UCSRnC cài đặt số lượng bit dữ liệu (kích cỡ chuỗi kí tự ) trong một khung truyền mà bộ thu nhận và bộ phát sử dụng

Bit 1 – RXB8n : bit 8 dữ liệu đến

RXB8n là bit dữ liệu thứ 9 của chuỗi kí tự đến khi mà hoạt động với các khung nối tiếp với 9 bit dữ liệu . Phải được đọc trước khi tiến hành việc đọc các bit thấp hơn từ UDRn

Bit 0 – TXB8n : bit 8 dữ liệu chuyển phát

TXB8n là bit dữ liệu thứ 9 trong chuỗi kí tự được chuyển phát khi mà hoạt động với các khung truyền nối tiếp với 9 bit dữ liệu . Phải được ghi trước khi việc ghi các bit thấp lên UDRn được tiến hành

### Thanh ghi C trạng thái và điều khiển USARTn – UCSRnC

Bit	7	6	5	4	3	2	1	0	UCSRnC
ReadWrite	-	UMSELn	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn	
Initial Value	0	0	0	0	0	1	1	0	

Chú ý rằng thanh ghi này không khả dụng trong chế độ tương thích với Atmega 103

Bit 7 – bit dự trữ

Bit này thì được dự trữ cho việc sử dụng trong tương lai . Để tương thích với các thiết bị trong tương lai , các bit này phải được ghi là 0 khi mà UCSRnC được ghi

Bit 6 – UMSELn : lựa chọn chế độ USART

Bit này lựa chọn giữa chế độ điều khiển đồng bộ và dị bộ

Table 77. UMSELn Bit Settings

UMSELn	Mode
0	Asynchronous Operation
1	Synchronous Operation

Bit 5:4 – UPMn1:0 : chế độ chẵn lẻ

Các bit này kích hoạt và cài đặt loại của sự phát và kiểm tra bit chẵn lẻ . Nếu được kích hoạt , bộ chuyển phát sẽ tự động sinh ra và gửi đi các bit dữ liệu chuyển phát chẵn lẻ trong mỗi khung truyền . Bộ thu nhận sẽ sinh ra 1 giá trị chẵn lẻ cho dữ liệu đến và so sánh nó với việc cài đặt UPMn0 . Nếu sự không tương ứng được tìm ra , cờ báo UPEn sẽ được cài đặt

**Table 78.** UPMn Bits Settings

UPMn1	UPMn0	Parity Mode
0	0	Disabled
0	1	(Reserved)
1	0	Enabled, Even Parity
1	1	Enabled, Odd Parity

Bit 3 – USBSn : lựa chọn bit stop

Bit này lựa chọn số lượng bit stop được chèn vào bởi bộ chuyển phát . Bộ thu nhận bỏ qua việc cài đặt này

**Table 79.** USBSn Bit Settings

USBSn	Stop Bit(s)
0	1-bit
1	2-bits

Bit 2:1 – UCSZn1:0 : kích cỡ chuỗi kí tự

Các bit UCSZn1:0 được kết hợp với UCSZn2 trong thanh ghi UCSRnB cài đặt số lượng của các bit dữ liệu (kích cỡ chuỗi kí tự) trong một khung truyền mà bộ thu nhận và bộ chuyển phát sử dụng

**Table 80.** UCSZn Bits Settings

UCSZn2	UCSZn1	UCSZn0	Character Size
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9-bit

Bit 0 – UCPOLn : cực xung nhịp

Bit này được chỉ sử dụng cho chế độ đồng bộ . Viết bit này là 0 khi chế độ dị bộ được sử dụng . bit UCPOLn cài đặt quan hệ giữa sự thay thế đầu ra dữ liệu và lấy mẫu đầu vào dữ liệu , và xung nhịp đồng bộ (XCKn)

**Table 81.** UCPOLn Bit Settings

UCPOLn	Transmitted Data Changed (Output of TxDn Pin)	Received Data Sampled (Input on RxDn Pin)
0	Rising XCKn Edge	Falling XCKn Edge
1	Falling XCKn Edge	Rising XCKn Edge

## Các thanh ghi Baud Rate USART – UBRRnL và UBRRnH

Bit	15	14	13	12	11	10	9	8	
	-	-	-	-	UBRRn[11:8]				UBRRnH
	UBRRn[7:0]								UBRRnL
Read/Write	7	6	5	4	3	2	1	0	
R/W	R	R	R	R	R/W	R/W	R/W	R/W	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

UBRRnH thì không khả dụng trong chế độ tương thích với Atmega 103

Bit 15:12 – Các bit dữ trữ

Các bit này được dữ trữ cho việc sử dụng trong tương lai . Để tương thích với các thiết bị trong tương lai , các bit này phải được viết là 0 khi mà UBRRnH được ghi

Bit 11:0 – UBRRn11:0 : thanh ghi baud rate USARTn

Đây là 1 12 thanh ghi bao gồm baud rate USARTn . UBRRnH bao gồm 4 bit có trọng số cao nhất , và UBRRnL bao gồm 8 bit có trọng số thấp nhất của baud rate USARTn . quá trình chuyển dữ liệu đang tiến hành bởi bộ chuyển phát và bộ thu sẽ bị làm hỏng nếu baud rate bị thay đổi . Việc viết UBRRnL sẽ khởi động một cập nhật trung gian của bộ đếm gộp trước Baudrate

## Ví dụ về việc cài đặt Baudrate

For standard crystal and resonator frequencies, the most commonly used baud rates for asynchronous operation can be generated by using the UBRR settings in Table 82. UBRR values which yield an actual baud rate differing less than 0.5% from the target baud rate, are bold in the table. Higher error ratings are acceptable, but the receiver will have less noise resistance when the error ratings are high, especially for large serial frames (see "Asynchronous Operational Range" on page 186). The error values are calculated using the following equation:

$$\text{Error}[\%] = \left( \frac{\text{BaudRate}_{\text{Closest Match}} - 1}{\text{BaudRate}} \right) \cdot 100\%$$

**Table 82.** Examples of UBRR Settings for Commonly Used Oscillator Frequencies

Baud Rate (bps)	$f_{osc} = 1.0000 \text{ MHz}$				$f_{osc} = 1.8432 \text{ MHz}$				$f_{osc} = 2.0000 \text{ MHz}$			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4k	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2k	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8k	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4k	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6k	0	8.5%	1	8.5%	1	0.0%	3	0.0%	1	8.5%	3	8.5%
76.8k	-	-	1	-18.6%	1	-25.0%	2	0.0%	1	-18.6%	2	8.5%
115.2k	-	-	0	8.5%	0	0.0%	1	0.0%	0	8.5%	1	8.5%
230.4k	-	-	-	-	-	-	0	0.0%	-	-	-	-
250k	-	-	-	-	-	-	-	-	-	-	0	0.0%
Max <sup>(1)</sup>	62.5 kbps		125 kbps		115.2 kbps		230.4 kbps		125 kbps		250 kbps	

1. UBRR = 0, Error = 0.0%

**Table 83.** Examples of UBRR Settings for Commonly Used Oscillator Frequencies

Baud Rate (bps)	$f_{osc} = 3.6864 \text{ MHz}$				$f_{osc} = 4.0000 \text{ MHz}$				$f_{osc} = 7.3728 \text{ MHz}$			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	95	0.0%	191	0.0%	103	0.2%	207	0.2%	191	0.0%	383	0.0%
4800	47	0.0%	95	0.0%	51	0.2%	103	0.2%	95	0.0%	191	0.0%
9600	23	0.0%	47	0.0%	25	0.2%	51	0.2%	47	0.0%	95	0.0%
14.4k	15	0.0%	31	0.0%	16	2.1%	34	-0.8%	31	0.0%	63	0.0%
19.2k	11	0.0%	23	0.0%	12	0.2%	25	0.2%	23	0.0%	47	0.0%
28.8k	7	0.0%	15	0.0%	8	-3.5%	16	2.1%	15	0.0%	31	0.0%
38.4k	5	0.0%	11	0.0%	6	-7.0%	12	0.2%	11	0.0%	23	0.0%
57.6k	3	0.0%	7	0.0%	3	8.5%	8	-3.5%	7	0.0%	15	0.0%
76.8k	2	0.0%	5	0.0%	2	8.5%	6	-7.0%	5	0.0%	11	0.0%
115.2k	1	0.0%	3	0.0%	1	8.5%	3	8.5%	3	0.0%	7	0.0%
230.4k	0	0.0%	1	0.0%	0	8.5%	1	8.5%	1	0.0%	3	0.0%
250k	0	-7.8%	1	-7.8%	0	0.0%	1	0.0%	1	-7.8%	3	-7.8%
0.5M	-	-	0	-7.8%	-	-	0	0.0%	0	-7.8%	1	-7.8%
1M	-	-	-	-	-	-	-	-	-	-	0	-7.8%
Max <sup>(1)</sup>	230.4 kbps		460.8 kbps		250 kbps		0.5 Mbps		460.8 kbps		921.6 kbps	

1. UBRR = 0, Error = 0.0%

**Table 84.** Examples of UBRR Settings for Commonly Used Oscillator Frequencies

Baud Rate (bps)	$f_{osc} = 8.0000 \text{ MHz}$				$f_{osc} = 11.0592 \text{ MHz}$				$f_{osc} = 14.7456 \text{ MHz}$			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	207	0.2%	416	-0.1%	287	0.0%	575	0.0%	383	0.0%	767	0.0%
4800	103	0.2%	207	0.2%	143	0.0%	287	0.0%	191	0.0%	383	0.0%
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4k	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2k	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8k	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4k	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%
57.6k	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8k	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2k	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%
230.4k	1	8.5%	3	8.5%	2	0.0%	5	0.0%	3	0.0%	7	0.0%
250k	1	0.0%	3	0.0%	2	-7.8%	5	-7.8%	3	-7.8%	6	5.3%
0.5M	0	0.0%	1	0.0%	-	-	2	-7.8%	1	-7.8%	3	-7.8%
1M	-	-	0	0.0%	-	-	-	-	0	-7.8%	1	-7.8%
Max <sup>(1)</sup>	0.5 Mbps		1 Mbps		691.2 kbps		1.3824 Mbps		921.6 kbps		1.8432 Mbps	

1. UBRR = 0, Error = 0.0%

**Table 85.** Examples of UBRR Settings for Commonly Used Oscillator Frequencies

Baud Rate (bps)	$f_{osc} = 16.0000 \text{ MHz}$				
	U2X = 0		U2X = 1		
	UBRR	Error	UBRR	Error	
2400	416	-0.1%	832	0.0%	
4800	207	0.2%	416	-0.1%	
9600	103	0.2%	207	0.2%	
14.4k	68	0.6%	138	-0.1%	
19.2k	51	0.2%	103	0.2%	
28.8k	34	-0.8%	68	0.6%	
38.4k	25	0.2%	51	0.2%	
57.6k	16	2.1%	34	-0.8%	
76.8k	12	0.2%	25	0.2%	
115.2k	8	-3.5%	16	2.1%	
230.4k	3	8.5%	8	-3.5%	
250k	3	0.0%	7	0.0%	
0.5M	1	0.0%	3	0.0%	
1M	0	0.0%	1	0.0%	
Max <sup>(1)</sup>	1 Mbps			2 Mbps	

1. UBRR = 0, Error = 0.0%

## XVIII. Giao diện hai dây tuần tự \_ Two – wire Serial Interface

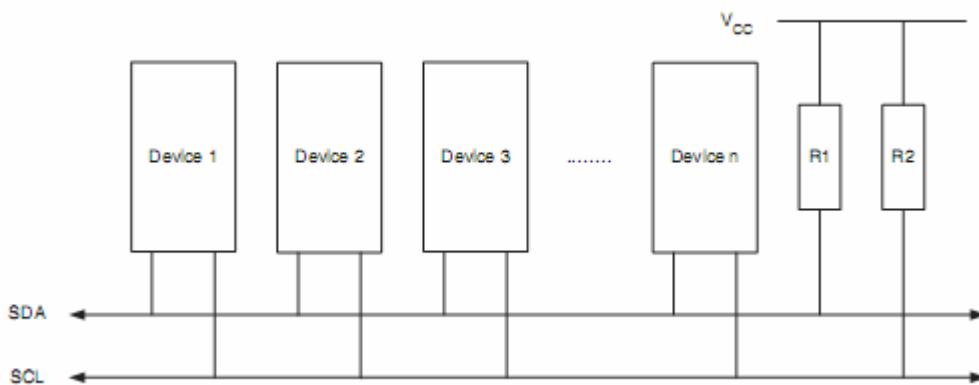
### Đặc điểm

- Đơn giản nhưng mạnh mẽ và giao diện truyền thông linh hoạt , chỉ cần thiết 2 đường bus
- Hỗ trợ cả chế độ điều khiển Master và Slave
- Thiết bị có thể hoạt động như một bộ chuyển phát hoặc một bộ thu
- Không gian địa chỉ 7-bit cho phép nâng lên 128 địa chỉ Slave khác nhau
- Hỗ trợ phân định chế độ Master – slave
- Nâng lên tốc độ chuyển dữ liệu là 400 kHz
- Các bộ điều khiển đều vào giới hạn tốc độ quay
- Mạch loại bỏ nhiễu loại bỏ đĩnh nhọn trên các đường bus
- Đầy đủ các địa chỉ slave lập trình được với sự hỗ trợ gọi chung
- Sự nhận ra địa chỉ gây ra đánh thức khi AVR trong các chế độ sleepmode

### Sự định nghĩa bus giao diện tuần tự hai dây

Giao diện tuần tự hai dây (TWI) là bộ công cụ lý tưởng cho các ứng dụng vi điều khiển thông thường . Giao thức TWI cho phép người thiết kế hệ thống liên kết trên 128 thiết bị khác nhau sử dụng chỉ 2 đường bus 2- hướng , 1 cho xung nhịp (SCL) và 1 cho dữ liệu (SDA) . Chỉ cần phần cứng bên ngoài để điều khiển bus như là 1 bộ điện trở pull-up đơn cho mỗi dây trong đường bus TWI . Tất cả các thiết bị được kết nối lên bus có địa chỉ riêng , và dẫn động cho bộ nhớ phân giải bên trong giao thức TWI

**Figure 86.** TWI Bus Interconnection



### Thuật ngữ của TWI

Các định nghĩa sau đây được bắt gặp thường xuyên trong phần này .

**Table 86. TWI Terminology**

Term	Description
Master	The device that initiates and terminates a transmission. The master also generates the SCL clock
Slave	The device addressed by a master
Transmitter	The device placing data on the bus
Receiver	The device reading data from the bus

## Liên kết điện

Như được miêu tả trong hình 86, cả hai đường bus đều được kết nối đến chân điện áp dương của nguồn cấp thông qua các điện trở Pull-up. Các bộ điều khiển bus của tất cả các thiết bị phù hợp với TWI là open – drain và open – collector. Điều này cài đặt 1 chức năng wried – AND cái mà cần thiết đến hoạt động của giao diện. Một mức logic thấp trên 1 đường bus TWI được sinh ra khi 1 hoặc nhiều hơn các đầu ra thiết bị TWI là 0. Một mức cao là đầu ra khi tất cả các thiết bị TWI 3 trạng thái của các đầu ra của chúng, sự cho phép của các điện trở pull-up để kéo lên mức cao. Chú ý rằng tất cả các thiết bị AVR được kết nối đến các bus TWI phải được cấp điện theo thứ tự để cho phép bắt cứ sự điều khiển các bus.

Số lượng của các thiết bị cái mà có thể kết nối tới bus chỉ bị giới hạn bằng bus điện dung giới hạn của  $400\text{pF}$  và không gian địa chỉ slave 7-bit. Một bảng thông số chi tiết của các đặc trưng của TWI được đưa trong phần “Two-wire Serial Interface Characteristics” trên trang 322. Hai sự cài đặt khác nhau của các đặc tính được đưa ra ở đó, 1 thông tin thích hợp cho các tốc độ bus dưới  $100\text{ kHz}$ , và 1 giá trị cho tốc độ bus trên  $400\text{kHz}$ .

## Chuyển dữ liệu vào dạng không chuyên

### Quá trình chuyển các bít

Mỗi bít dữ liệu được chuyển trên bus TWI được kèm theo một xung trên đường truyền xung nhịp. Cấp của đường dữ liệu phải ổn định khi đường xung nhịp ở mức cao. Chỉ ngoại trừ nguyên tắc này cho việc phát ra các điều kiện khởi động và dừng.

**Table 87. TWI Bit Rate Prescaler**

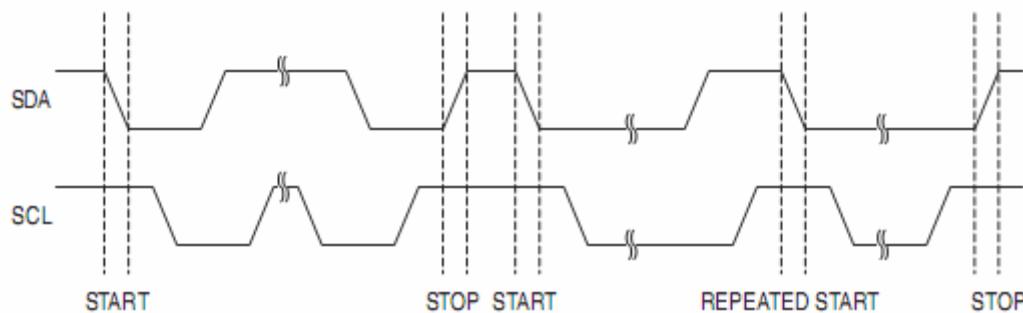
TWPS1	TWPS0	Prescaler Value
0	0	1
0	1	4
1	0	16
1	1	64

## Các điều kiện khởi động và dừng

Master khởi tạo và kết thúc một quá trình chuyển dữ liệu. Quá trình chuyển dữ liệu được khởi tạo khi Master đưa ra một điều kiện bắt đầu trên bus, và nó bị kết thúc khi master đưa ra một điều kiện stop. Giữa một điều kiện start và stop, bus đang được

xét đến bận, và không có Master nào khác đang cố gắng điều khiển bus dữ liệu. Một trường hợp đặc biệt xuất hiện khi một điều kiện start mới được đưa ra giữa một điều kiện start và stop cũ. Điều này được tham chiếu như là một điều kiện start lặp lại ( REPEATED START ) và được sử dụng khi Master được yêu cầu để khởi tạo một quá trình chuyển phát mới mà thiếu đi quá trình điều khiển relinquishing của bus. Sau khi một ( REPEATED START ), bus được xét đến bận cho đến khi có điều kiện stop tiếp theo. Đây là điều kiện riêng để xử lý bit start, và vì vậy bit start được sử dụng để miêu tả cả hai trạng thái START và ( REPEATED START ) cho các phần còn lại trong tài liệu này, trừ khi có chú ý khác. Như giàn đồ biên dưới, các điều kiện start và stop được ký hiệu bằng cách thay đổi mức của dòng SDA, khi mà dòng SCL ở mức cao.

**Figure 88. START, REPEATED START and STOP Conditions**



### Dạng đóng gói địa chỉ

Tất cả các gói địa chỉ được truyền trên bus TWI là 9 bit dài, bao gồm 7 bit địa chỉ, một bit điều khiển đọc ghi, và một bit nhận biết. Nếu bit đọc ghi được cài đặt, một quá trình đọc được tiến hành, nói cách khác một quá trình viết nên được tiến hành. Khi một slaver nhận ra rằng nó đang được đánh địa chỉ. Nó nên được nhận biết rằng việc kéo SDA ở mức thấp trong bit thứ 9 SCL ( ACK ). Nếu như slaver được đánh địa chỉ đang bận hoặc một vài lý do khác có thể không phục vụ truy vấn của Master. Dòng SDA nên được chuyển đến mức cao trong chu kỳ xung nhịp ACK . Master sau đó có thể chuyển một điều kiện stop, một ( REPEATED START ) để khởi tạo một quá trình chuyển phát mới. Một gói địa chỉ thì bao gồm một địa chỉ slaver và một bit đọc hoặc ghi được gọi là SLA+R và SLA+W một cách tương ứng.

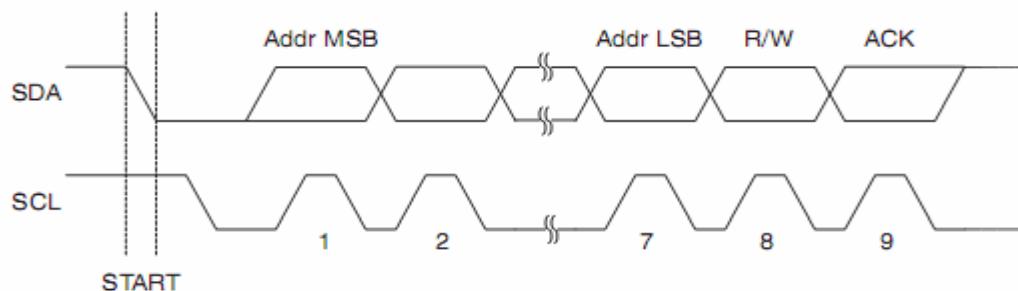
MSB của byte địa chỉ được truyền đầu tiên. Địa chỉ slaver đầu tiên có thể được đặt một cách tự do bởi người thiết kế nhưng địa chỉ 0000000 là dữ trữ cho các phép gọi chung.

Khi một phép gọi chung được đưa ra, tất cả các slaver nên đáp ứng bằng cách kéo dòng SDA trong chu kỳ ACK . Một sự gọi chung được sử dụng khi mà một master yêu cầu để chuyển một tin nhắn giống nhau đến các slaver nối tiếp trong hệ thống. Khi gọi địa chỉ chung được theo bởi một bit ghi được truyền trên bus truyền, tất cả các slaver cài đặt trên sự gọi chung đó sẽ kéo dòng SDA xuống mức thấp trong chu kỳ hỏi. Các gói dữ liệu tiếp theo của sự gọi chung đó được nhận bởi tất cả các slaver mà hiểu biết được các tín hiệu đó. Chú ý rằng quá trình các lệnh gọi địa chỉ chung đó được kèm theo

bởi một bit đọc thì không có nghĩa, điều này sẽ gây ra hư hỏng nếu các slaver nối tiếp được bắt đầu truyền các dữ liệu khác nhau.

Tất cả các địa chỉ của dạng 1111 xxx nên được dũ trữ cho các tính năng trong tương lai.

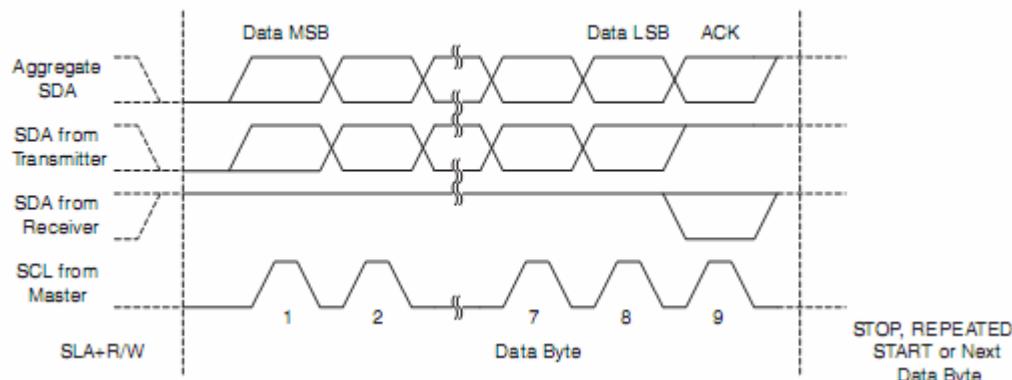
**Figure 89. Address Packet Format**



## Dạng gói dữ liệu

Tất cả các gói dữ liệu được truyền trên bus TWI có độ dài 9 bit bao gồm 1 byte và một bit nhận biết. Trong suốt một quá trình chuyển dữ liệu, master sinh ra một xung nhịp và các điều kiện start stop, trong khi nhận một đáp ứng cho quá trình nhận biết tín hiệu đến. Một tín hiệu nhận biết (ACK) được nhận biết bằng cách kéo xung tín hiệu trên dòng SDA ở mức thấp trong suốt chu kỳ SCL thứ 9. Nếu như bộ thu nhận chuyển dòng SDA lên mức cao, một NACK được nhận ra. Khi bộ nhận nhận được bus cuối cung, hoặc một vài lý do không thể nhận thêm bất cứ bus nào nữa, nó nên ở dạng byet cuối cùng. Bằng cách chuyển NACK sau byet cuối cùng. MSB của byet dữ liệu thì được chuyển đầu tiên.

**Figure 90. Data Packet Format**



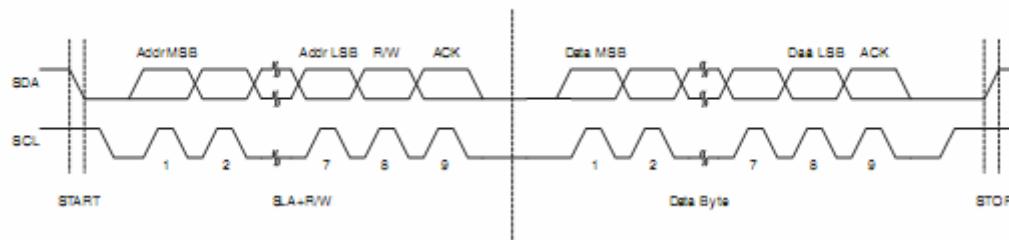
## Việc nối các địa chỉ và đóng gói dữ liệu trong một quá trình chuyển phát.

Một quá trình chuyển phát cơ bản bao gồm một điều kiện start, một SLA+R/W, một hoặc nhiều gói dữ liệu và một điều kiện stop. Một tin nhắn trống, bao gồm một điều kiện start kèm theo bởi một điều kiện stop một cách đồng thời. Chú ý rằng Wired-ANDing của dòng SCR có thể được sử dụng để cài đặt handshaking giữa master và slaver. Slaver có thể mở rộng chu kỳ dòng thấp SCL bằng cách kéo dòng SCL ở mức thấp. điều này thì có ích nếu tốc độ xung nhịp cài đặt bởi Master là quá nhanh cho

slaver, hoặn slaver cần thêm thời gian cho tiến trình giữa các quá trình truyền dữ liệu. Việc mở rộng slaver trong chu kỳ SCL thấp sẽ không ảnh hưởng đến SCL cao, cái mà xác định bởi master. Như một nối tiếp, slaver có thể giảm tốc độ chuyển dữ liệu TWI bằng prolonging chu kỳ SCL.

Hình 91 chỉ ra một quá trình chuyển phát thông thường. các byte dữ liệu nối tiếp có thể được truyền giữa SLA+R/W và điều kiện stop, phụ thuộc vào giao thức phần mềm được cài đặt bởi phần mềm ứng dụng.

Figure 91. Typical Data Transmission

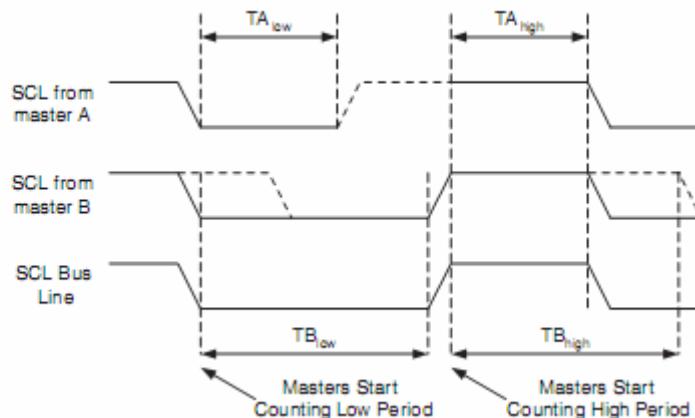


### Các hệ thống bus nhiều master.

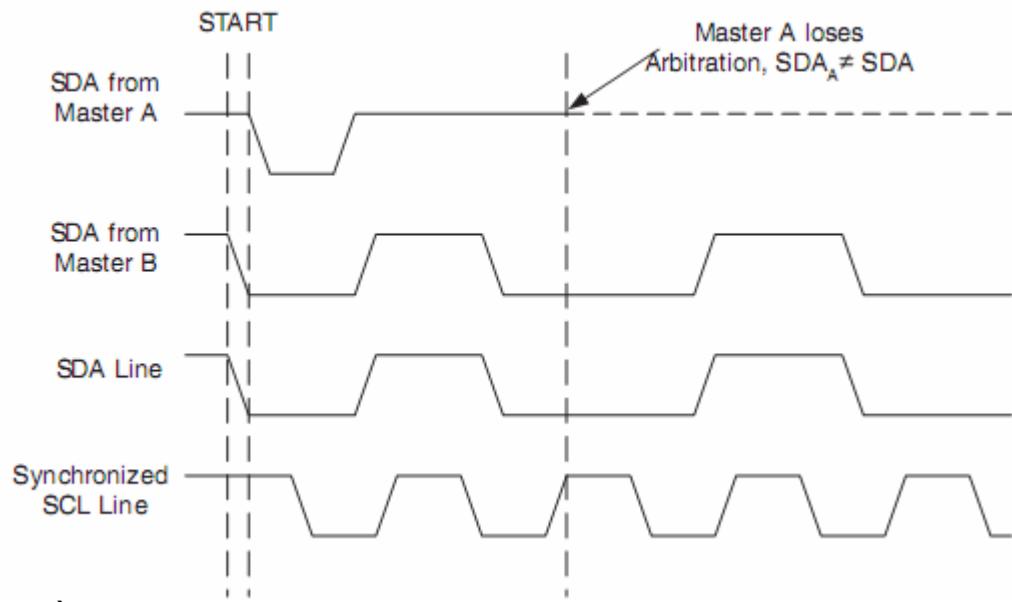
Giao thức TWI cho phép bus hệ thống với nhiều master nối tiếp các trường hợp đặc biệt là có thể xảy ra để đảm bảo rằng quá trình chuyển phát được thực thi như thông thường, nếu 2 hoặc nhiều master khởi tạo một quá trình truyền phát tại cùng một thời điểm. 2 vấn đề này sinh trong các hệ thống nhiều master:

- Một thuật toán phải được cài đặt chỉ cho phép một số các master để hoàn thành quá trình chuyển phát. Tất cả các master khác đang truyền dữ liệu, khi chúng phát hiện ra rằng dữ liệu của chúng bị mất trong các quá trình được lựa chọn. Các quá trình lựa chọn này được gọi là arbitration. Khi một master đang truyền dữ liệu phát hiện ra rằng nó bị mất quá trình arbitration, nó sẽ ngay lập tức chuyển mạch sang chế độ slaver để kiểm tra quá trình đánh địa chỉ của nó bởi các master có ưu thế hơn. Chính xác là, các master vừa khởi động quá trình truyền phát tại cùng một thời điểm nên không thể dò thay các slaver ( ví dụ: dữ liệu đang được chuyển trên bus không phải bị gián đoạn).
- Các master riêng biệt có thể sử dụng các tần số SCL riêng biệt. Một sơ đồ được chia ra để đồng bộ hóa các xung nối tiếp từ tất cả các master, để bắt đầu một quá trình truyền phát trong một lockstep fashion. Điều này sẽ khắc phục một quá trình arbitration.

Wired-ANDing của các đường bus được sử dụng để khắc phục các vấn đề này. Các xung nhịp nối tiếp từ tất cả các master sẽ được Wired-ANDing, yielding một kết nối xung nhịp với một chu kỳ cao bằng một trong số các master với chu kỳ cao ngắn nhất. Chu kỳ thấp của xung nhịp kết nối thì bằng với chu kỳ thấp của master với chu kỳ thấp dài nhất. Chú ý rằng tất cả các master lắng nghe dòng lệnh SCL việc bắt đầu một cách hiệu quả để đếm các chu kỳ time-out thấp và cao SCL khi các dòng kết nối SCL ở mức cao hoặc thấp, một cách tương ứng.

**Figure 92.** SCL Synchronization between Multiple Masters

Arbitration được mang ra ngoài bởi tất cả các master một cách liên tục quan sát dòng SDA sau khi xuất đầu ra dữ liệu. Nếu giá trị được đọc từ dòng SDA thì không tương ứng với giá trị mà master có ở đầu ra, nó vừa bị mất Arbitration. Chú ý rằng một master chỉ có thể bị mất Arbitration khi các đầu ra của nó ở giá trị SDA cao trong khi các đầu ra của các master khác ở giá trị thấp việc mất dữ liệu của master nên chuyển sang chế độ slaver ngay lập tức, việc kiểm tra nếu nó đang được đánh địa chỉ bởi master có quyền ưu tiên cao hơn. Dòng SDA nên được chuyển lên cao nhưng các master bị mất dữ liệu thì được cho phép để phát ra một tín hiệu xung nhịp cho đến khi kết thúc dữ liệu hiện hành hoặc gói địa chỉ hiện hành. Arbitration sẽ tiếp tục cho đến khi chỉ còn lại một master, điều này có thể tạo ra nhiều bit hơn. Nếu các master nối tiếp đang cố gắng để đánh địa chỉ slaver giống nhau, Arbitration sẽ tiếp tục diễn ra trong gói dữ liệu.

**Figure 93.** Arbitration Between two Masters

Chú ý rằng Arbitration thì không được cho phép giữa.

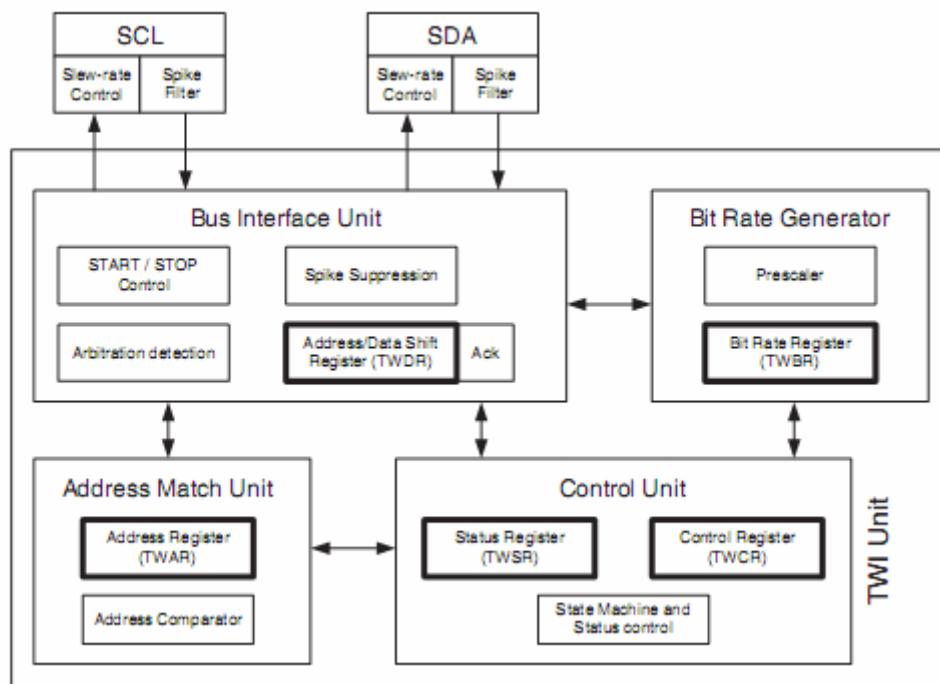
- Một điều kiện REPEATED START .
- Một điều kiện stop và một bit dữ liệu
- REPEATED START và điều kiện stop

Đây là sự tương ứng của phần mềm người dùng để đảm bảo rằng illegal Arbitration không bao giờ xuất hiện. điều này đảm bảo rằng trong một hệ thống nhiều master, tất cả các sự chuyển phát dữ liệu phải sử dụng các thành phần của SLA+R/W và các gói dữ liệu. trong các dạng từ khác: tất cả các quá chuyển phát phải bao gồm các số giống nhau của các gói dữ liệu, nói cách khác kết quả của Arbitration là không xác định.

### Tổng quan của module TWI

Module TWI được bao gồm rất nhiều module bên trong như được chỉ ra trong hình 94. Tất cả các thanh ghi được vẽ bằng một nét mảnh thì được truy nhập thông qua bus dữ liệu AVR.

**Figure 94. Overview of the TWI Module**



### Các chân Scl và SDA

Các chân giao diện AVR TWI với các chân nghỉ hệ thống MCU đầu ra của bộ điều khiển bao gồm một bộ giới hạn slew-rate để mà để phù hợp với các đặc điểm kỹ thuật của TWI. Các cổng đầu ra bao gồm một bộ phận loại trừ sung gỡ bỏ các tín hiệu sung ngắn hơn 50 ns. Chú ý rằng các xung pull-áp bên trong AVR có thể được kích hoạt bằng việc cài đặt các bít cổng PORT tương ứng lên các chân SCL và SDA như được diễn tả trong phần cổng I/O. Các xung pull-áp bên trong có thể trong một vài hệ thống triệt tiêu là cần thiết cho các thiết bị bên ngoài.

## Bộ phận máy phát bitrate

Bộ phận này điều khiển chu kỳ của SCL khi hoạt động trong chế độ Master . Chu kì SCL được điều khiển bằng việc cài đặt trong thanh ghi bit Rate TWI (TWBR) và các bit của bộ đếm gộp trước trong thanh ghi trạng thái TWI (TWSR) . Slave hoạt động không phụ thuộc vào bit Rate hoặc việc cài đặt bộ đếm gộp trước , nhưng tần số xung nhịp trong chế độ slave phải được dưới 16 lần cao hơn tần số SCL . Chú ý rằng các slave có thể mở rộng chu kì thấp SCL , vì vậy việc làm giá trị trung bình chu kì xung nhịp bus TWI . Tần số SCL được phát ra theo công thức dưới đây :

$$\text{SCL frequency} = \frac{\text{CPU Clock frequency}}{16 + 2(\text{TWBR}) \cdot 4^{\text{TWPS}}}$$

- TWBR = giá trị của thanh ghi Bit Rate TWI
- TWPS = giá trị của các bit của bộ đếm gộp trước trong thanh ghi trạng thái TWI

Chú ý rằng : giá trị điện trở Pull-up nên được lựa chọn theo tần số xung nhịp của và dung lượng tải trên bus . xem bảng 133 trên trang 322 cho giá trị của điện trở pull

## Thành phần giao diện bus

Thành phần này bao gồm các thanh ghi shift địa chỉ và dữ liệu (TWDR,một bộ điều khiển STAR STOP và phần cứng dò ARBITRATION . TWDR bao gồm các bai địa chỉ và các bai dữ liệu được chuyển phát hoặc các bai địa chỉ hoặc các bai dữ liệu được nhận thêm vào đó 8 bit TWDR , bộ phận giao diện bus cũng bao gồm một thanh ghi chứa bit NACK để truyền hoặc nhận dữ liệu . thanh ghi NACK này thì không thể truy nhập trực tiếp bằng các phần mềm ứng dụng . Tuy nhiên , khi đang nhận nó có thể được cài đặt hoặc bị xóa bằng việc điều khiển thanh ghi điều khiển TWI . Khi trong chế độ chuyển phát , giá trị của bit (N)ACK đã nhận có thể xác định bằng giá trị trong TWSR

Bộ điều khiển START/STOP thì phải chịu trách nhiệm cho quá trình phát và quá trình dò của các điều kiện START , và REPEATED START , và STOP . Bộ điều khiển START/STOP thì có thể dò các điều kiện START và STOP dù khi MCU của AVR đang trong một chế độ ngủ nào đó , việc kích hoạt MCU để đánh thức nếu được đánh địa chỉ bởi Master

Nếu TWI vừa được khởi tạo 1 quá trình chuyển phát như là master , phần cứng dò tìm kiểm định tiếp tục quan sát quá trình truyền dữ liệu đang cố gắng để xác định nếu sự kiểm định của nó đang tiến hành . Nếu TWI vừa mất một sự kiểm định , bộ phận điều khiển thì được truyền dữ liệu . Một hành động đúng sau đó có thể hành động và một code trạng thái tương ứng có thể được sinh ra .

## Bộ phận ghép địa chỉ

Bộ phận ghép địa chỉ kiểm tra nếu như các byte địa chỉ đã nhận tương ứng với địa chỉ 7 bit trong thanh ghi địa chỉ TWI (TWAR). Nếu như bit kích hoạt sự nhận diện lệnh gọi chung TWI (TWGCE) trong TWAR được viết là 1, tất cả các bit địa chỉ đến cũng sẽ được so sánh lại với địa chỉ gọi chung. Trên một ghép địa chỉ, bộ phận điều khiển có thể được thông báo, việc cho phép các hành động đúng được xảy ra. TWI có thể hoặc không thể nhận ra địa chỉ của nó phụ thuộc vào việc cài đặt trong TWCR. Bộ phận ghép địa chỉ có thể so sánh địa chỉ khi MCU của AVR trong chế độ ngủ, việc kích hoạt MCU để đánh thức nếu được đánh địa chỉ bởi một Master. Nếu các ngắt khác (ví dụ INT0) xuất hiện trong suốt quá trình ghép địa chỉ Power-down TWI và đánh thức CPU, TWI bỏ qua quá trình điều khiển và trở về trạng thái idle của nó. Nếu điều này gây ra bát cứ vấn đề nào, đảm bảo rằng ghép địa chỉ TWI thì chỉ được kích hoạt ngắt khi đang truy nhập vào chế độ Power-down.

## Bộ phận điều khiển

Bộ phận điều khiển giám sát bus TWI và sinh ra những đáp ứng tương ứng để cài đặt trong thanh ghi điều khiển TWI (TWCR). Khi một sự kiện đang cần thiết phải chú ý đến của các ứng dụng xuất hiện trong bus TWI, cờ ngắt TWI (TWINT) được xác nhận. Trong chu kỳ xung nhịp tiếp theo, thanh ghi trạng thái TWI (TWSR) được cập nhật với một mã trạng thái riêng biệt của sự kiện. TWSR chỉ bao gồm các thông tin trạng thái xác đáng khi các cờ ngắt TWI được xác nhận. Tại tất cả các thời điểm khác TWSR bao gồm 1 mã trạng thái đặc biệt đang hiển thị rằng không có thông tin trạng thái xác đáng là khả dụng. Chỉ cần TWINT được cài đặt, đường SCL được giữ ở mức thấp. Điều này cho phép phần mềm ứng dụng để hoàn thành nhiệm vụ của nó trước khi cho phép quá trình truyền TWI tiếp tục.

Cờ TWINT được cài đặt trong các trường hợp dưới đây

- Sau khi TWI vừa được chuyển phát 1 điều kiện START/REPEATED START
- Sau khi TWI vừa được chuyển SLA+R/W
- Sau khi TWI vừa được chuyển 1 byte địa chỉ
- Sau khi TWI vừa mất quá trình kiểm định
- Sau khi TWI vừa được đánh địa chỉ bằng các địa chỉ Slave riêng hoặc gọi địa chỉ chung
- Sau khi TWI vừa nhận 1 byte dữ liệu
- Sau khi điều kiện STOP hoặc REPEATED START vừa được nhận trong khi vẫn đánh địa chỉ như là một slave
- Khi một lỗi bus vừa được xuất hiện dù cho 1 điều kiện START và STOP không hợp lệ

## Sự miêu tả thanh ghi TWI

### Thanh ghi Bit Rate TWI – TWBR

Bit	7	6	5	4	3	2	1	0	TWBR
ReadWrite	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bit 7..0 thanh ghi Bit Rate TWI

TWBR lựa chọn tỉ lệ chia cho máy phát tốc độ bit Rate . Máy phát Bit Rate thì một bộ chia tần số cái mà sinh ra xung nhịp SCL trong các chế độ Master . Xem “Bit Rate Generator Unit “ trên trang 204 về việc tính toán bit Rate

### Thanh ghi điều khiển TWI – TWCR

Bit	7	6	5	4	3	2	1	0	TWCR
ReadWrite	R/W	R/W	R/W	R/W	R	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TWCR thì được sử dụng để điều khiển hoạt động của TWI .Nó được sử dụng để kích hoạt TWI , để khởi tạo một sự truy nhập master bằng việc đặt một điều kiện START lên bus , để sinh ra 1 bộ thu nhận biết được sinh ra 1 điều kiện STOP , và điều khiển việc dừng của bus trong khi dữ liệu được viết lên bus được ghi lên thanh ghi TWDR . Nó cũng hiển thị 1 sự xung đột nếu dữ liệu được cố gắng ghi đè lên TWDR trong khi thanh ghi không thể truy nhập .

Bit 7 – TWINT : cờ ngắt TWI

Bit này được cài đặt bằng phần cứng khi TWI vừa hoàn thành công việc hiện hành của nó và đợi phần mềm ứng dụng đáp ứng . Nếu bit I trong SREG và TWIE trong TWCR được cài đặt , MCU sẽ nhảy tới vec tơ ngắt TWI . Trong khi cờ TWINT được cài đặt thì chu kỳ SCL ở mức thấp bị kéo dài

Cờ TWINT phải bị xóa bằng phần mềm bằng việc viết mức logic 1 lên nó . Chú ý rằng cờ này không tự động bị xóa bằng phần cứng khi đang thực thi chương trình con phục vụ ngắt . Cũng chú ý rằng việc xóa cờ này bắt đầu hoạt động của TWI 1, vì vậy tất cả các sự truy nhập tới thanh ghi địa chỉ TWI (TWAR), và thanh ghi trạng thái TWI (TWSR), và thanh ghi dữ liệu TWI (TWDR) phải được hoàn thành trước khi xóa cờ này

Bit 6 – TWEA : bit nhận biết kích hoạt TWI

TWEA điều khiển sự sinh ra của xung nhận biết . Nếu bit TWEA được viết là 1 , xung ACK được sinh ra trên bus TWI nếu các điều kiện sau đây được đáp ứng :

- địa chỉ slave của thiết bị vừa được nhận đến
- một lệnh gọi chung vừa được nhận , trong khi bit TWGCE trong TWAR được cài đặt
- một byte địa chỉ vừa được nhận trong bộ thu nhận Master và chế độ thu nhận slave

bằng việc viết TWEA là 0 , thiết bị có thể ngắt kết nối ảo khỏi bus TWI một cách tạm thời . Sự nhận diện địa chỉ sau đó có thể khôi phục lại bằng việc viết bit TWEA là 1 một lần nữa

Bit 5 – TWSTA : bit điều kiện khởi động TWI

Úng dụng viết bit TWSTA là 1 khi mà nó muốn trở thành một master trên bus TWI . Phần cứng TWI kiểm tra nếu bus là khả dụng , và sinh ra một điều kiện START trên bus nếu như nó tự do . Tuy nhiên , nếu bus không tự do thì TWI phải đợi cho đến khi điều kiện STOP được dò thấy , và sau đó sinh ra 1 điều kiện START mới để kháng nghị về trạng thái của bus Master. TWSTA phải được xóa bằng phần mềm khi mà điều kiện START vừa được truyền đi

Bit 4 – TWSTO : bit điều kiện STOP TWI

Việc viết bit TWSTO lên 1 trong chế độ Master sẽ sinh ra 1 điều kiện STOP trên bus TWI . Khi điều kiện STOP được thực thi trên bus , bit TWSTO bị xóa 1 cách tự động . Trong chế độ Slave , việc cài đặt bit TWSTO có thể được sử dụng để khôi phục từ một điều kiện lỗi . Điều này sẽ không sinh ra một điều kiện STOP , nhưng TWI trở lại như là một chế độ slave chưa được đánh địa chỉ (a well-defined unaddressed )và giải phóng đường SCL và SDA lên trạng thái trở kháng cao

Bit 3 – TWWC – cờ báo viết xung đột viết TWI

TWWC cờ này được cài đặt khi đang cố gắng viết lên thanh ghi dữ liệu TWI – TWDR khi mà TWINT ở mức thấp . Cờ này bị xóa bằng việc viết lên thanh ghi TWDR khi mà TWINT ở mức cao

Bit 2 – TWEN : bit kích hoạt TWI

TWEN kích hoạt hoạt động TWI và kích hoạt giao diện TWI . Khi TWEN được viết là lên 1 , TWI điều khiển qua các chân I/O được kết nối đến các chân SCL và SDA , việc kích hoạt bộ giới hạn slew-rate và bộ lọc nhiễu . Nếu bit này được viết là 0 , TWI bị tắt và tất cả các quá trình chuyển TWI đều kết thúc bất chấp bất cứ hoạt động điều khiển nào đang tiến hành

Bit 1 – Res : bit dự trữ

Bit này là một bit dự trữ và sẽ luôn được đọc là 0

Bit 0 – TWIE : kích hoạt ngắt TWI

Khi bit này được viết là 1 , và bit I trong thanh ghi SREG được cài đặt , yêu cầu ngắt TWI sẽ được kích hoạt chỉ khi mà cờ TWINT ở mức cao

## Thanh ghi trạng thái TWI – TWSR

Bit	7	6	5	4	3	2	1	0	TWSR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	1	1	1	1	1	0	0	0	

Bit 7..3 – TWS : trạng thái TWI

Đây là 5 bit phản ánh trạng thái của logic TWI và bus hai dây tuần tự . Các mã trạng thái khác nhau được miêu tả sau trong phần này . Chú ý rằng giá trị đọc từ TWSR chứa cả hai giá trị trạng thái 5bit và giá trị bộ đếm gộp trước 2bit . Người thiết kế ứng dụng nên che các bit bộ đếm gộp trước lên 0 khi đang kiểm tra các bit trạng thái . Điều này tạo ra việc kiểm tra trạng thái độc lập với việc cài đặt bộ đếm gộp trước . Điều này giống như được sử dụng trong datasheet này , trừ phi được chú ý khác .

Bit 2 – Res : bit dự trữ

Bit này là bit dự trữ và luôn được đọc là 0

Bit 1..0 – TWPS : các bit đếm gộp trước TWI

Các bit này có thể được đọc và viết , và điều khiển đếm gộp trước bit rate

**Table 87. TWI Bit Rate Prescaler**

TWPS1	TWPS0	Prescaler Value
0	0	1
0	1	4
1	0	16
1	1	64

Để tính toán các bit rate , xem “bộ phận phát Bit Rate” trên trang 204 . Giá trị của TWPS1 ..0 được sử dụng trong công thức

### Thanh ghi dữ liệu TWI – TWDR

Bit	7	6	5	4	3	2	1	0	TWDR
Read/Write	R/W								
Initial Value	1	1	1	1	1	1	1	1	

Trong chế độ chuyển phát , TWDR chứa byte kế tiếp được truyền phát . Trong chế độ nhận , TWDR bao gồm byte cuối cùng được nhận . Nó không thể được viết trong khi TWI không ở trong quá trình đang di chuyển 1 byte . Điều này xuất hiện khi mà cờ báo ngắt TWI (TWINT)được cài đặt bằng phần cứng .Chú ý rằng thanh ghi dữ liệu không thể được khởi tạo bằng người sử dụng trước khi ngắt đầu tiên xuất hiện . Dữ liệu trong TWDR còn lại ổn định chỉ cần TWINT được cài đặt . Trong khi dữ liệu được di chuyển ra ngoài , dữ liệu trên bus được di chuyển vào trong một cách đồng thời . TWDR luôn chứa byte cuối cùng đưa ra trên bus, ngoại trừ sau khi có một sự đánh thức từ trạng thái ngủ bằng ngắt TWI . Trong trường hợp này , thành phần của TWDR là không xác định. Trong trường hợp của việc mất sự kiểm định trên bus , không có dữ liệu nào bị mất trong quá trình chuyển từ Master xuống Slave . Việc điều khiển của bit ACK được điều khiển một cách tự động bằng logic TWI , CPU không thể truy nhập vào bit ACK một cách trực tiếp

Bit 7...0 – TWD : thanh ghi dữ liệu TWI

Tám bit này cấu thành byte dữ liệu kế tiếp được chuyển phát , hoặc byte dữ liệu cuối cùng được nhận trên bus 2 dây tuần tự

## Thanh ghi địa chỉ TWI (Slave) – TWAR

Bit	7	6	5	4	3	2	1	0	TWAR
Read/Write	R/W								
Initial Value	1	1	1	1	1	1	1	0	

TWAR nên được tải với 7 bit địa chỉ slave (trong 7 bit có trọng số cao nhất của TWAR ) lên cái mà TWI sẽ đáp ứng khi được lập trình như là một bộ chuyển phát và thu nhận slave , và không cần thiết trong các chế độ master . Trong hệ thống nhiều master , TWAR phải được cài đặt trong master cái mà có thể được đánh địa chỉ như các slave bởi các master khác

LSB của TWAR được sử dụng để kích hoạt quá trình nhận biết của địa chỉ gọi chung (\$00) . Có một bộ so sánh đã ghép địa chỉ cái mà tìm kiếm địa chỉ slave (hoặc địa chỉ gọi chung được kích hoạt) trong địa chỉ nối tiếp đã nhận . Nếu 1 ghép tương ứng được tìm thấy , một yêu cầu ngắt được sinh ra.

Bit 7..1 –TWA : thanh ghi địa chỉ TWI (slave )

7 bit này cấu thành nên địa chỉ slave của bộ phận TWI

Bit 0 – TWGCE : bit kích hoạt nhận diện gọi chung TWI

Nếu được cài đặt , bit này kích hoạt quá trình nhận diện của 1 lệnh gọi chung được đưa ra qua bus 2 dây tuần tự

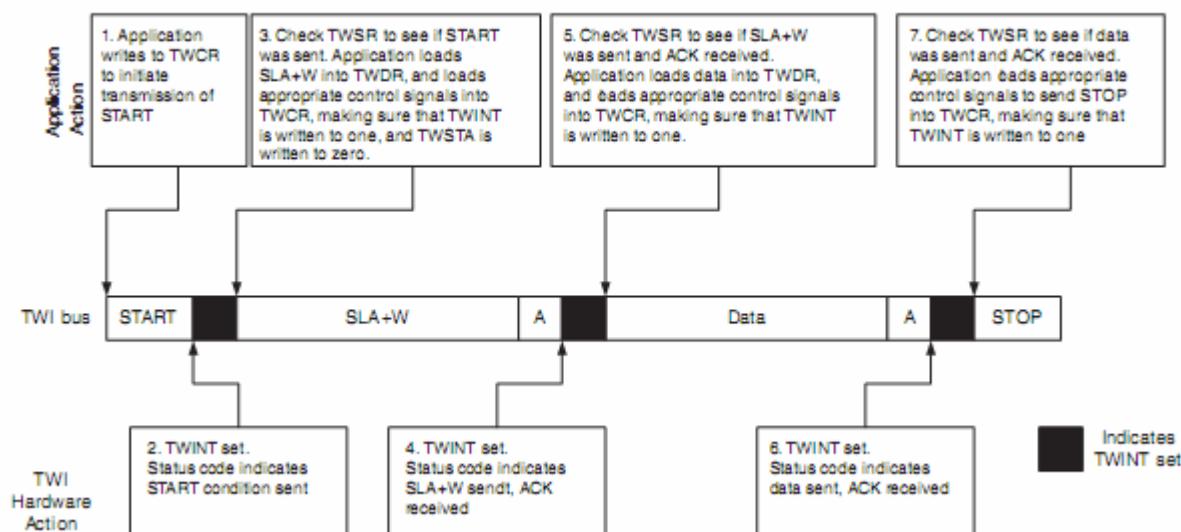
## Việc sử dụng TWI

AVR TWI là byte đã định hướng và ngắt cơ sở . Các ngắt được ban hành sau khi tất cả các bus sự kiện , giống như sự thu nhận của 1 byte hoặc quá trình truyền phát của một điều kiện START . Bởi vì TWI là một ngắt cơ sở , phần mềm ứng dụng là tự do để tiếp tục các hoạt động khác trong suốt 1 quá trình truyền byte TWI . Chú ý rằng bit kích hoạt ngắt TWI là (TWIE) trong TWCR cùng với bit kích hoạt ngắt chung trong thanh ghi SREG cho phép ứng dụng để quyết định hoặc không có sự xác nhận của cờ báo TWINT nên sinh ra một yêu cầu ngắt . Nếu bit TWIE bị xóa , ứng dụng phải được hỏi vòng cờ TWINT để dò tìm hành động trên bus TWI

Khi cờ báo TWINT được xác nhận , TWI vừa hoàn thành một quá trình điều khiển và đang đợi ứng dụng đáp ứng . Trong trường hợp này thanh ghi trạng thái TWI (TWSR) chứa một giá trị đang hiển thị trạng thái hiện hành của bus TWI . Phần mềm ứng dụng sau đó có thể quyết định cách mà TWI nên tiến hành trong bus TWI kế tiếp bằng việc xử lý TWCR và các thanh ghi TWDR

Hình 95 là một ví dụ mẫu của cách mà ứng dụng có thể giao diện với phần cứng TWI . Trong ví dụ này , một master muốn chuyển 1 byte dữ liệu đơn đến 1 slave . Sự miêu tả này là khá tóm tắt , một sự diễn tả chi tiết hơn sẽ ở phần sau của phần này . 1 chương trình mẫu về sự cài đặt muốn tiến hành thì cũng được đưa ra .

Figure 95. Interfacing the Application to the TWI in a Typical Transmission



1. Bước đầu tiên trong quá trình truyền phát TWI thì để truyền một điều kiện START . Điều này được thực hiện bởi việc viết một giá trị xác định vào trong TWCR , sự hướng dẫn cho phần cứng TWI để truyền một điều kiện START . Cái mà giá trị để viết thì được miêu tả sau . Tuy nhiên , nó thì quan trọng cái mà bit TWINT được cài đặt trong giá trị được viết . Việc viết là một lên bit TWINT xóa cờ . TWI sẽ không bắt đầu bắt cứ quá trình điều khiển nào chỉ cần bit TWINT trong thanh ghi TWCR được cài đặt . Ngay lập tức sau khi ứng dụng vừa được xóa TWINT , TWI sẽ khởi tạo quá trình truyền dữ liệu của điều kiện START
2. Khi điều kiện START vừa được truyền đi , cờ báo TWINT trong thanh ghi TWCR được cài đặt , và TWSR được cập nhật với mã trạng thái hiển thị rằng điều kiện START vừa được gửi thành công.
3. Phần mềm ứng dụng nên kiểm tra giá trị của TWSR bây giờ , để đảm bảo rằng điều kiện START vừa được chuyển thành công. Nếu TWSR hiển thị theo cách khác , phần mềm ứng dụng có thể tạo ra vài hành động đặc biệt , giống như việc gọi 1 chương trình con bị lỗi . Giá định rằng mã trạng thái thì được mong đợi , ứng dụng phải tải SLA+W vào trong TWDR. Nhớ rằng TWDR được sử dụng cho cả địa chỉ và dữ liệu. Sau khi TWDR vừa được tải với SLA+W xác định , 1 giá trị xác định phải được viết lên TWCR , hướng dẫn phần cứng TWI truyền SLA+W đưa vào trong TWDR . Giá trị để viết thì được xác định sau . Tuy nhiên , quan trọng là bit TWINT được cài đặt trong giá trị được ghi . Việc viết là 1 lên TWINT xóa cờ . TWI sẽ không bắt đầu bắt cứ hoạt động chỉ cần bit TWINT trong TWCR được cài đặt . Ngay lập tức sau khi ứng dụng vừa xóa TWINT , TWI sẽ khởi tạo quá trình truyền dữ liệu của gói địa chỉ .
4. Khi gói dữ liệu vừa được chuyển , cờ báo TWINT trong TWCR được cài đặt , và TWSR được cập nhật với mã trạng thái hiển thị rằng gói địa chỉ vừa được

- chuyển thành công . Mã trạng thái sẽ phản ánh rằng 1 slave nhận biết gói dữ liệu hoặc là không
5. phần mềm ứng dụng nên kiểm tra giá trị của TWSR ngay bây giờ , để đảm bảo rằng gói địa chỉ đã được chuyển thành công, và giá trị của bit ACK đã như mong đợi . Nếu TWSR hiển thị theo cách khác , phần mềm ứng dụng có thể tạo ra vài hành động đặc biệt , giống như lệnh gọi một chương trình con lỗi . Giá định rằng mã trạng thái thì như mong đợi , ứng dụng phải tải 1 gói dữ liệu vào trong TWDR . Sau đó , một giá trị xác định phải được viết lên TWCR , hướng dẫn phần cứng TWI để chuyển một gói dữ liệu đưa vào trong TWDR. Giá trị mà để viết lên thì được miêu tả trong phần sau . Tuy nhiên , cái quan trọng là TWINT được cài đặt trong giá trị được viết . Việc viết là 1 lê TWINT xóa cờ . TWI sẽ không khởi động bất cứ quá trình nào chỉ cần TWINT trong TWCR được cài đặt . Ngay lập tức sau khi ứng dụng vừa xóa TWINT , TWI sẽ khởi tạo một quá trình truyền của một gói dữ liệu
  6. Khi một gói dữ liệu vừa được chuyển , cờ TWINT trong thanh ghi TWCR được cài đặt , và TWSR được cập nhật mã trạng thái hiển thị mà gói dữ liệu vừa được chuyển thành công . Mã trạng thái cũng sẽ phản ánh 1 sự nhận biết slave là gói hoặc không
  7. Phần mềm ứng dụng nên được kiểm tra giá trị của TWSR , để đảm bảo rằng gói dữ liệu vừa được chuyển phát thành công , và cái mà giá trị của bit ACK đã được mong đợi . Nếu TWSR hiển thị theo cách khác , phần mềm ứng dụng có thể tạo ra một vài hành động đặc biệt , giống như việc gọi một chương trình con bị lỗi . Giá định rằng mã trạng thái như là đã mong đợi , ứng dụng phải viết một giá trị xác định lên TWCR , hướng dẫn phần cứng để truyền điều kiện STOP . Giá trị được viết thì được miêu tả sau . Tuy nhiên , quan trọng là bit TWINT được cài đặt trong giá trị đã được ghi . Việc viết là 1 lê TWINT xóa cờ báo . TWI sẽ không bắt đầu bắt cứ hành động nào chỉ cần TWINT trong TWCR được cài đặt . Ngay lập tức sau khi ứng dụng vừa xóa TWINT , TWI sẽ khởi tạo quá trình chuyển phát của điều kiện STOP . CHú ý rằng TWINT thì không được cài đặt sau một điều kiện STOP vừa được gửi

Mặc dù ví dụ này là đơn giản , nó cũng chỉ ra nguyên tắc được bao hàm trong tất cả các quá trình truyền TWI . Những điều này có thể được liệt kê chi tiết như bên dưới :

- Khi mà TWI vừa hoàn thành 1 hoạt động và chờ đợi đáp ứng của ứng dụng , cờ TWINT được cài đặt . Đường SCL được kéo xuống mức thấp cho đến khi TWINT bị xóa .
- Khi mà cờ TWINT được cài đặt , người sử dụng phải cập nhật tất cả các thanh ghi TWI với giá trị xác đáng cho chu kỳ bus TWI . Như một ví dụ , TWDR phải được tải với giá trị được truyền trong chu kỳ bus tiếp theo .
- Sau khi thanh ghi TWI cập nhật và các phần mềm ứng dụng đang chờ khác làm nhiệm vụ vừa hoàn thành , TWCR được viết . Khi việc viết TWCR , bit

TWINT nên được cài đặt . Việc viết là 1 lện TWINT xóa cờ . TWI sau đó sẽ bắt đầu thực thi cái mà quá trình điều khiển đã xác định bằng việc cài đặt TWCR

Bên dưới là một đoạn mã chương trình C và Assembly của ví dụ được đưa ra . chú ý rằng mã bên dưới được xác định bằng nhiều định nghĩa vừa được tạo cho ví dụ bằng việc sử dụng các file bao gồm .

	Assembly Code Example	C Example	Comments
1	ldi r16, (1<<TWINT)   (1<<TWSTA)   (1<<TWEN) out TWCR, r16	TWCR = (1<<TWINT)   (1<<TWSTA)   (1<<TWEN)	Send START condition
2	wait1: in r16,TWCR sbrs r16,TWINT rjmp wait1	while (!(TWCR & (1<<TWINT))) ;	Wait for TWINT flag set. This indicates that the START condition has been transmitted
3	in r16,TWSR andi r16, 0xF8 ori r16, START brne ERROR  ldi r16, SLA_W out TWDR, r16 ldi r16, (1<<TWINT)   (1<<TWEN) out TWCR, r16	if ((TWSR & 0xF8) != START) ERROR();  TWDR = SLA_W; TWCR = (1<<TWINT)   (1<<TWEN);	Check value of TWI Status Register. Mask prescaler bits. If status different from START go to ERROR  Load SLA_W into TWDR Register. Clear TWINT bit in TWCR to start transmission of address
4	wait2: in r16,TWCR sbrs r16,TWINT rjmp wait2	while (!(TWCR & (1<<TWINT))) ;	Wait for TWINT flag set. This indicates that the SLA+W has been transmitted, and ACK/NACK has been received.
5	in r16,TWSR andi r16, 0xF8 ori r16, MT_SLA_ACK brne ERROR  ldi r16, DATA out TWDR, r16 ldi r16, (1<<TWINT)   (1<<TWEN) out TWCR, r16	if ((TWSR & 0xF8) != MT_SLA_ACK) ERROR();  TWDR = DATA; TWCR = (1<<TWINT)   (1<<TWEN);	Check value of TWI Status Register. Mask prescaler bits. If status different from MT_SLA_ACK go to ERROR  Load DATA into TWDR Register. Clear TWINT bit in TWCR to start transmission of data
6	wait3: in r16,TWCR sbrs r16,TWINT rjmp wait3	while (!(TWCR & (1<<TWINT))) ;	Wait for TWINT flag set. This indicates that the DATA has been transmitted, and ACK/NACK has been received.
7	in r16,TWSR andi r16, 0xF8 ori r16, MT_DATA_ACK brne ERROR  ldi r16, (1<<TWINT)   (1<<TWEN)   (1<<TWSTO) out TWCR, r16	if ((TWSR & 0xF8) != MT_DATA_ACK) ERROR();  TWCR = (1<<TWINT)   (1<<TWEN)   (1<<TWSTO);	Check value of TWI Status Register. Mask prescaler bits. If status different from MT_DATA_ACK go to ERROR  Transmit STOP condition

Note: For I/O registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

## Các chế độ truyền phát dữ liệu

TWI có thể hoạt động trong một trong 4 chế độ chính . Chúng có tên là Master Transmitter (MT) , Master Receiver (MR) , Slave Transmitter (ST) , Slave Receiver (SR). Một vài trong số những chế độ đó có thể được sử dụng trong những ứng dụng giống nhau . Như là một ví dụ , TWI có thể sử dụng chế độ MT để viết dữ liệu vào trong TWI EEPROM , chế độ MR để đọc dữ liệu trở lại từ EEPROM . Nếu các Master

khác được đưa ra trong hệ thống , một vài trong số chúng có thể chuyển dữ liệu đến TWI , và sau đó chế độ SR sẽ được sử dụng . Nó là phần mềm ứng dụng mà quyết định chế độ nào được sử dụng

Phân tiếp theo miêu tả các chế độ này . Mã trạng thái có thể được miêu tả cùng với các hình vẽ chi tiết của quá trình truyền dữ liệu trong mỗi chế độ . Các hình vẽ này bao gồm các kí hiệu viết tắt bên dưới :

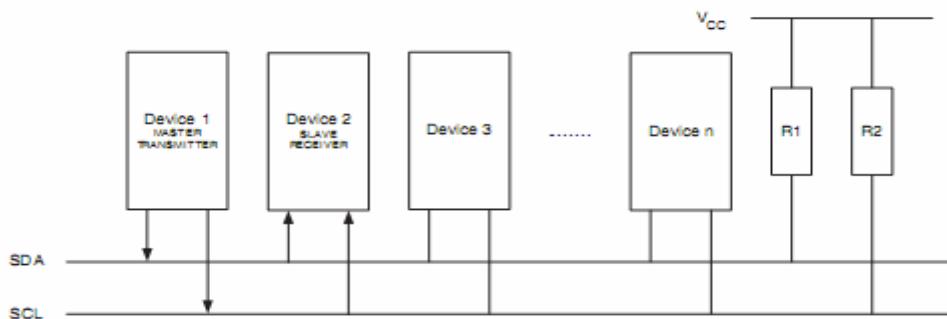
- S : điều kiện START
- Rs : điều kiện REPEATED START
- R : bit đọc (mức cao tại SDA )
- W : Bit viết (mức thấp tại SDA )
- A : bit nhận biết (mức thấp tại SDA )
- A : bit không nhận biết (mức cao tại SDA)
- Data : byte dữ liệu 8 bit
- P : điều kiện STOP
- SLA : địa chỉ Slave

Trong hình 97 đến hình 103 , vòng tròn được sử dụng để hiển thị rằng cờ TWINT đã được cài đặt . Số của vòng tròn chỉ ra mã trạng thái được giữ trong TWSR , với các bit bộ đếm gộp trước được che bởi 0 . Tại các điểm này , hành động phải được thực hiện bằng ứng dụng để tiếp tục hoặc hoàn thành quá trình truyền TWI . Quá trình chuyển TWI được hoàn cho đến khi cờ TWINT bị xóa bằng phần mềm .

Khi cờ TWINT được cài đặt , mã trạng thái trong TWSR được sử dụng để xác định thích ứng với hành động của phần mềm . Cho mỗi trạng mã trạng thái , hành động cần thiết của phần mềm và chi tiết của quá trình truyền dữ liệu nối tiếp bên dưới được đưa ra trong bảng 88 đến 91 . Chú ý rằng các bit của bộ đếm gộp trước được che là 0 trong các bảng này

### **Chế độ bộ chuyển phát Master (Master Transmitter Mode )**

Trong chế độ bộ chuyển phát Master , 1 số các bit của các byte dữ liệu được chuyển tới một bộ thu nhận slave (xem hình 96) . Để truy nhập vào chế độ Master , một điều kiện START phải được chuyển phát đi . Dạng của các gói địa chỉ kế tiếp được xác định chế độ chuyển phát Master hoặc chế độ thu nhận Master được truy nhập . Nếu SLA+W được chuyển , chế độ MT được truy nhập , nếu SLA+R được chuyển phát , chế độ MR được truy nhập . Tất cả các mã trạng thái được nói đến trong phần này giả định rằng các bit bộ đếm gộp trước là 0 hoặc được che bởi 0 .

**Figure 96. Data Transfer in Master Transmitter Mode**

Một điều kiện START được gửi bằng cách viết giá trị dưới đây lên TWCR

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	1	0	X	1	0	X

TWEN phải được cài đặt để kích hoạt giao diện 2 dây tuần tự , TWSTA phải được ghi là 1 để chuyển một điều kiện START và TWINT phải được viết là 1 để xóa cờ TWINT . TWI sau đó sẽ kiểm tra bus 2 dây tuần tự và sinh ra một điều kiện START chỉ cần bus trống rảnh rồi . Sau khi một điều kiện START vừa được chuyển đi , cờ báo TWINT được cài đặt bằng phần cứng , và mã trạng thái trong TWSR sẽ là \$08 (xem bảng 88) . Để truy nhập vào chế độ MT thì SLA+W phải được chuyển đi . Điều này được thực hiện bằng cách viết SLA+W lên TWDR . Sau đó bit TWINT nên được xóa (bằng việc viết nó là 1) để tiếp tục truyền dữ liệu . Điều này được hoàn thành bằng việc viết giá trị kế tiếp lên TWCR

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	0	0	X	1	0	X

Khi SLA+W vừa hoàn thành việc chuyển phát , 1 gói dữ liệu nên được chuyển phát . Điều này được thực hiện bằng cách viết byte dữ liệu lên TWDR . TWDR chỉ phải viết khi TWINT ở mức cao . Nếu không , sự truy cập sẽ bị hủy bỏ , và bit xung đột viết (TWWC) sẽ được cài đặt trong thanh ghi TWCR . Sau khi cập nhật TWDR , bit TWINT nên bị xóa (bằng việc viết nó là 1 ) để tiếp tục quá trình chuyển phát . Điều này được hoàn thành bằng việc viết giá trị sau lên TWCR

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	0	0	X	1	0	X

Sơ đồ này được lặp lại cho đến khi byte cuối cùng vừa được gửi và quá trình chuyển phát kết thúc bằng việc sinh ra 1 điều kiện STOP hoặc lặp lại điều kiện STOP . 1 điều kiện STOP được sinh ra bằng việc viết giá trị dưới đây lên TWCR :

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	0	1	X	1	0	X

Một điều kiện REPEATED START được sinh ra bằng việc viết giá trị dưới đây lên TWCR

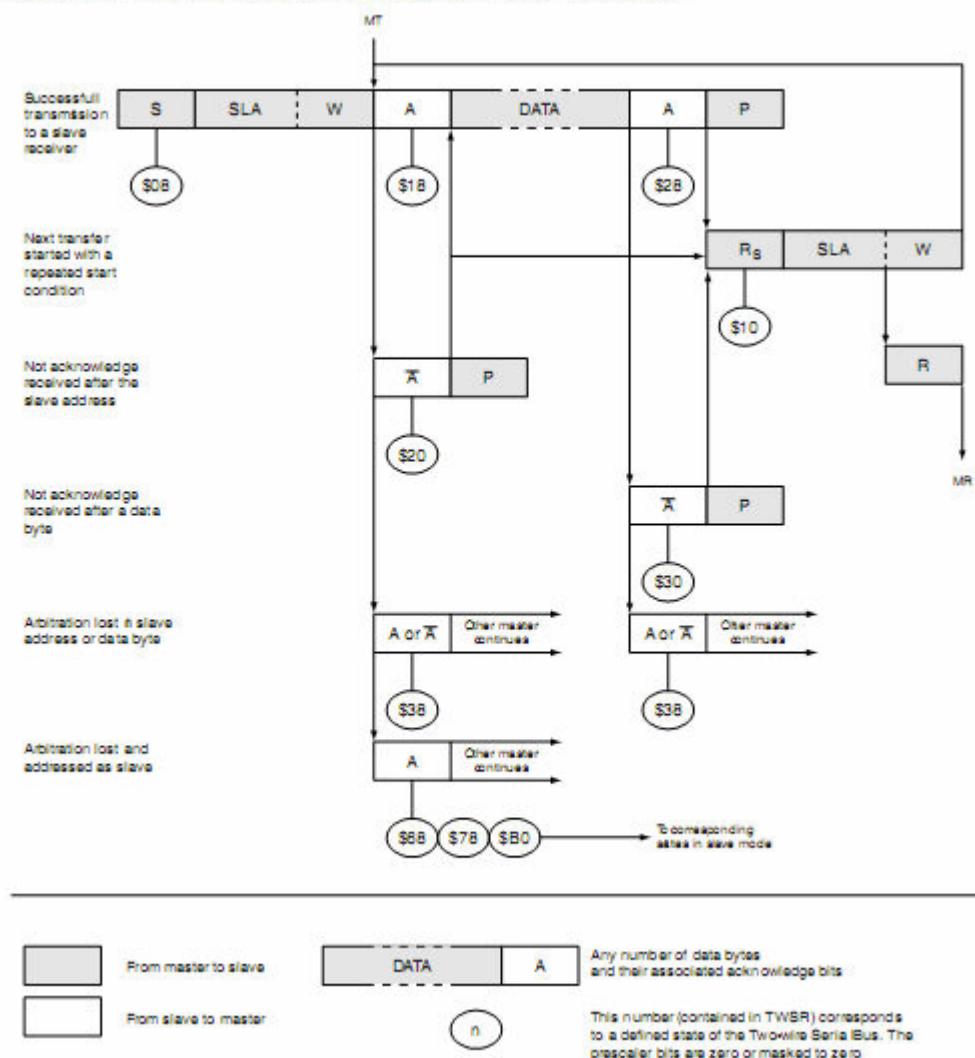
TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	1	0	X	1	0	X

Sau khi một sự lặp lại điều kiện START (trạng thái \$10) giao diện 2 dây tuần tự có thể truy nhập slave giống nhau lại , hoặc 1 slave mới thiếu quá trình truyền một điều

kiện STOP . Lặp lại START kích hoạt master để chuyển mạch giữa các slave , chế độ máy phát Master và chế độ bộ thu nhận Master thiêu mêt việc điều khiển của bus.

Table 88. Status Codes for Master Transmitter Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the Two-wire Serial Bus and Two-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/TWCR	STA	STO	TWINT	TWEA	
\$08	A START condition has been transmitted	Load SLA+W	0	0	1	X	SLA+W will be transmitted; ACK or NOT ACK will be received
\$10	A repeated START condition has been transmitted	Load SLA+W or Load SLA+R	0 0	0 0	1 1	X X	SLA+W will be transmitted; ACK or NOT ACK will be received SLA+R will be transmitted; Logic will switch to master receiver mode
\$18	SLA+W has been transmitted; ACK has been received	Load data byte or No TWDR action or No TWDR action or	0 1 0	0 0 1	1 1 1	X X X	Data byte will be transmitted and ACK or NOT ACK will be received Repeated START will be transmitted STOP condition will be transmitted and TWSTO flag will be reset STOP condition followed by a START condition will be transmitted and TWSTO flag will be reset
\$20	SLA+W has been transmitted; NOT ACK has been received	Load data byte or No TWDR action or No TWDR action or	0 1 0	0 0 1	1 1 1	X X X	Data byte will be transmitted and ACK or NOT ACK will be received Repeated START will be transmitted STOP condition will be transmitted and TWSTO flag will be reset STOP condition followed by a START condition will be transmitted and TWSTO flag will be reset
\$28	Data byte has been transmitted; ACK has been received	Load data byte or No TWDR action or No TWDR action or	0 1 0	0 0 1	1 1 1	X X X	Data byte will be transmitted and ACK or NOT ACK will be received Repeated START will be transmitted STOP condition will be transmitted and TWSTO flag will be reset STOP condition followed by a START condition will be transmitted and TWSTO flag will be reset
\$30	Data byte has been transmitted; NOT ACK has been received	Load data byte or No TWDR action or No TWDR action or	0 1 0	0 0 1	1 1 1	X X X	Data byte will be transmitted and ACK or NOT ACK will be received Repeated START will be transmitted STOP condition will be transmitted and TWSTO flag will be reset STOP condition followed by a START condition will be transmitted and TWSTO flag will be reset
\$38	Arbitration lost in SLA+W or data bytes	No TWDR action or No TWDR action	0 1	0 0	1 1	X X	Two-wire Serial Bus will be released and not addressed slave mode entered A START condition will be transmitted when the bus becomes free

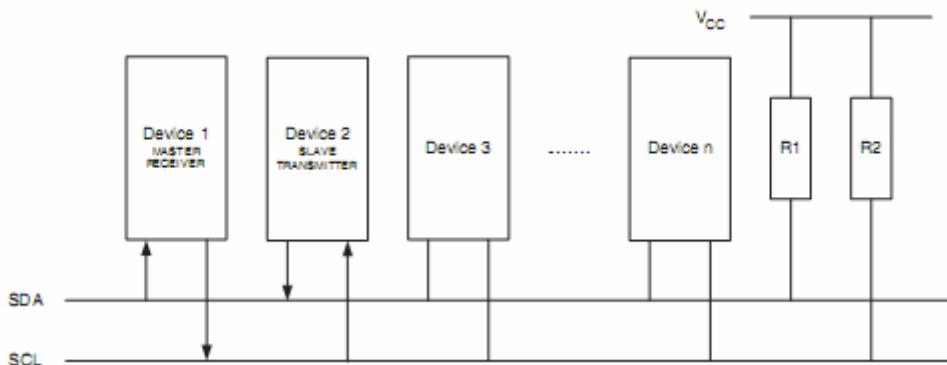
**Figure 97.** Formats and States in the Master Transmitter Mode

### Chế độ bộ thu nhận Master (Master Receiver Mode )

Trong chế độ bộ thu nhận Master , 1 số của các byte dữ liệu nhận từ 1 bộ chuyển phát slave (xem hình 98) . Để truy nhập vào chế độ Master , 1 điều kiện START phải được chuyển phát .. Dạng của các gói địa chỉ kế tiếp được xác định chế độ chuyển phát Master hoặc chế độ thu nhận Master được truy nhập . Nếu SLA+W được chuyển , chế độ MT được truy nhập , nếu SLA+R được chuyển phát , chế độ MR được truy nhập . Tất cả các mã trạng thái được nói đến trong phần này giả định rằng các bit bộ đếm gộp trước là 0 hoặc được che bởi 0 .

Hình 98 . chuyển dữ liệu trong chế độ bộ thu nhận Master

Figure 98. Data Transfer in Master Receiver Mode



Một điều kiện START được gửi bằng việc viết giá trị dưới đây lên TWCR :

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	1	0	X	1	0	X

TWEN phải được viết là 1 để kích hoạt giao diện 2 dây tuần tự , TWSTA phải được ghi là 1 để chuyển điều kiện START và TWINT phải được cài đặt để xóa cờ TWINT . TWI sau đó sẽ kiểm tra bus 2 dây tuần tự và sinh ra một điều kiện START chỉ cần bus trở nên tự do . Sau khi điều kiện START vừa được chuyển đi , cờ TWINT được cài đặt bằng phần cứng , và mã trạng thái trong TWSR sẽ là \$08 (xem bảng 88). Để truy nhập vào chế độ MR , SLA+R phải được chuyển đi . Điều này được thực hiện bằng việc viết SLA+R lên TWDR . Sau đó bit TWINT nên bị xóa (bằng việc viết nó là 1) để tiếp tục chuyển dữ liệu . Điều này được thực hiện bằng việc viết giá trị dưới đây lên TWCR :

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	0	0	X	1	0	X

Khi SLA+R vừa được chuyển và một bit nhận biết vừa được nhận , TWINT được cài đặt lại và 1 số của các mã trạng thái trong thanh ghi TWSR là có thể . Mã trạng thái có thể trong chế độ Master là \$38 , \$40 , \$48 . Hành động thích đáng được xảy ra khi cho mỗi một trong các mã trạng thái thanh ghi được liệt kê chi tiết trong bảng 97 . Tín hiệu đã nhận có thể được đọc từ thanh ghi TWDR khi mà cờ TWINT được cài đặt ở mức cao bởi phần cứng . Sơ đồ này được lập lại cho đến khi byte cuối cùng được nhận . Sau khi byte cuối cùng vừa được nhận , MR nên được thông báo ST bằng việc gửi 1 NACK sau khi byte dữ liệu cuối cùng được nhận. Sự chuyển phát phải được kết thúc bằng việc sinh ra một điều kiện STOP hoặc 1 điều kiện START được lặp lại . Một điều kiện STOP được sinh ra bằng việc viết giá trị dưới đây lên TWCR

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	0	1	X	1	0	X

Một điều kiện REPEATED START được sinh ra bằng việc viết giá trị dưới đây lên TWCR :

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	1	X	1	0	X	1	0	X

Sau khi lặp lại điều kiện START (trạng thái \$10) giao diện 2 dây tuần tự có thể truy nhập lại vào slave giống nhau , hoặc 1 slave mới thiếu sự truyền phát 1 điều kiện

STOP . Lập lại điều kiện START kích hoạt master để chuyển mạch giữa các Slave , chế độ truyền phát Master và thu nhận Master thiêu vắng sự mất dữ liệu điều khiển trên bus

Figure 99. Formats and States in the Master Receiver Mode

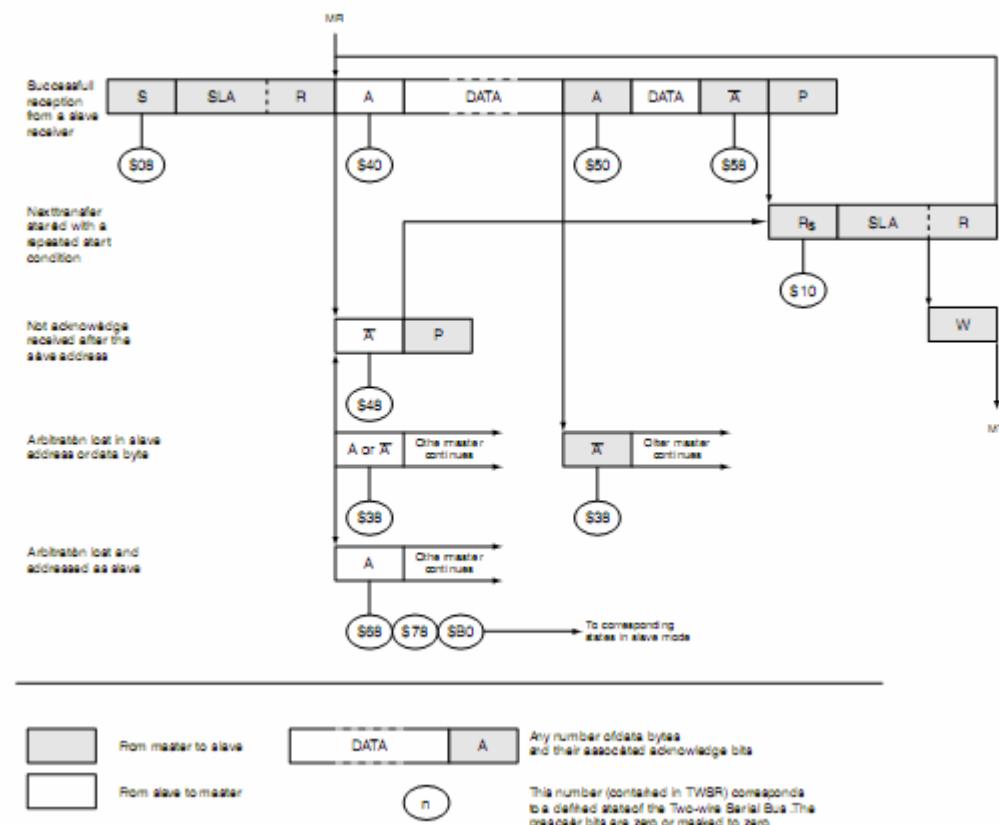


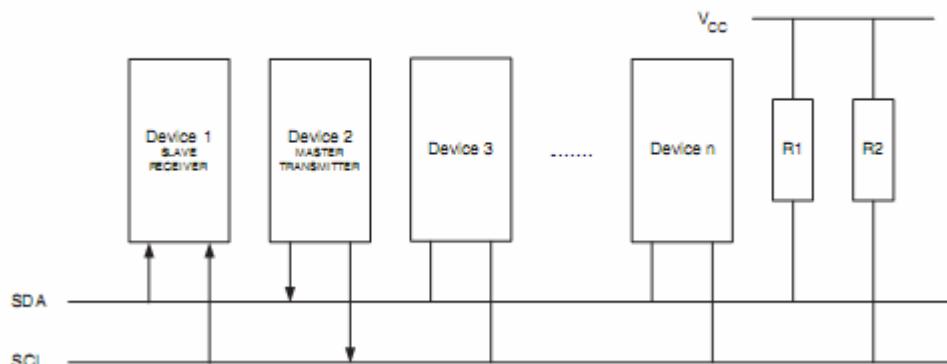
Table 89. Status Codes for Master Receiver Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the Two-wire Serial Bus and Two-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware	
		To/Tfrom TWDR		To TWCR				
		STA	STO	TWINT	TWEA			
\$08	A START condition has been transmitted	Load SLA+R	0	0	1	X	SLA+R will be transmitted. ACK or NOT ACK will be received.	
\$10	A repeated START condition has been transmitted	Load SLA+R or Load SLA+W	0 0	0 1	1	X	SLA+R will be transmitted. ACK or NOT ACK will be received. SLA+W will be transmitted. Logic will switch to master transmitter mode.	
\$38	Arbitration lost in SLA+R or NOT ACK bit	No TWDR action or No TWDR action	0 1	0 1	1	X	Two-wire Serial Bus will be released and not addressed slave mode will be entered. A START condition will be transmitted when the bus becomes free.	
\$40	SLA+R has been transmitted; ACK has been received	No TWDR action or No TWDR action	0 0	0 1	1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$48	SLA+R has been transmitted; NOT ACK has been received	No TWDR action or No TWDR action or No TWDR action	1 0 1	0 1 1	1	X X	Repeated START will be transmitted. STOP condition will be transmitted and TWSTO flag will be reset. STOP condition followed by a START condition will be transmitted and TWSTO flag will be reset.	
\$50	Data byte has been received; ACK has been returned	Read data byte or Read data byte	0 0	0 1	1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$58	Data byte has been received; NOT ACK has been returned	Read data byte or Read data byte or Read data byte	1 0 1	0 1 1	1	X X	Repeated START will be transmitted. STOP condition will be transmitted and TWSTO flag will be reset. STOP condition followed by a START condition will be transmitted and TWSTO flag will be reset.	

## Chế độ bộ thu nhận slave (slave Receiver Mode )

Trong chế độ bộ thu nhận Slave , 1 số của các byte dữ liệu được nhận từ 1 bộ chuyển phát master(xem hình 100) . Tất cả các mã trạng thái được nói đến trong phần này giả định rằng các bit bộ đếm gộp trước là 0 hoặc bị che bởi 0 .

**Figure 100.** Data Transfer in Slave Receiver Mode



Để khởi tạo chế độ bộ thu nhận Slave , TWAR và TWCR phải được khởi tạo như bên dưới :

TWAR value	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
Device's Own Slave Address								

Hơn 7 bit thì được đánh địa chỉ tới cái mà giao diện 2 dây tuần tự sẽ đáp ứng khi được đánh địa chỉ bởi master . Nếu như LSB được cài đặt , TWI sẽ đáp ứng để đánh địa chỉ gọi chung (\$00) , nói cách khác nó sẽ bỏ qua địa chỉ gọi chung .

TWCR value	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
	0	1	0	0	0	1	0	X

TWEN phải được ghi là 1 để kích hoạt TWI . bit TWEA phải được ghi là 1 để kích hoạt sự thura nhận của địa chỉ slave của riêng thiết bị hoặc địa chỉ gọi chung . TWSTA và TWSTO phải được ghi là 0 .

Khi TWAR và TWCR vừa được khởi tạo , TWI đợi cho đến khi nó được đánh địa chỉ bởi slave riêng của nó (hoặc địa chỉ lệnh gọi chung nếu được kích hoạt ) được theo bởi bit định hướng dữ liệu . Nếu bit định hướng là 0 (ghi ) , TWI sẽ hoạt động trong chế độ SR , nói cách khác chế độ ST được truy nhập . Sau khi địa chỉ slave riêng của nó và bit ghi vừa được nhận , cờ báo TWINT được cài đặt và một mã trạng thái có hiệu lực có thể đọc từ TWSR . Mã trạng thái được sử dụng để xác định hành động thích đáng của phần mềm . Hành động thích đáng của phần mềm tạo ra trong mỗi mã trạng thái được nêu chi tiết trong bảng 90 . Chế độ bộ nhận Slave cũng có thể được truy nhập nếu như quá trình giám định bị mất trong khi TWI trong chế độ master (xem các trạng thái \$68 và \$78)

Nếu bit TWEA được reset trong suốt một quá trình chuyển phát , TWI sẽ phản hồi một “Not Acknowledge” “1” tới SDA sau khi byte dữ liệu tiếp theo được nhận . Điều này có thể được sử dụng để hiển thị rằng Slave thì không thể nhận bất cứ byte nào nhiều hơn .Trong khi TWEA là 0 , TWI không nhận ra địa chỉ slave riêng của nó . Tuy nhiên , bus 2 dây tuần tự vẫn được quan sát và sự nhận diện địa chỉ có thể khôi phục tại

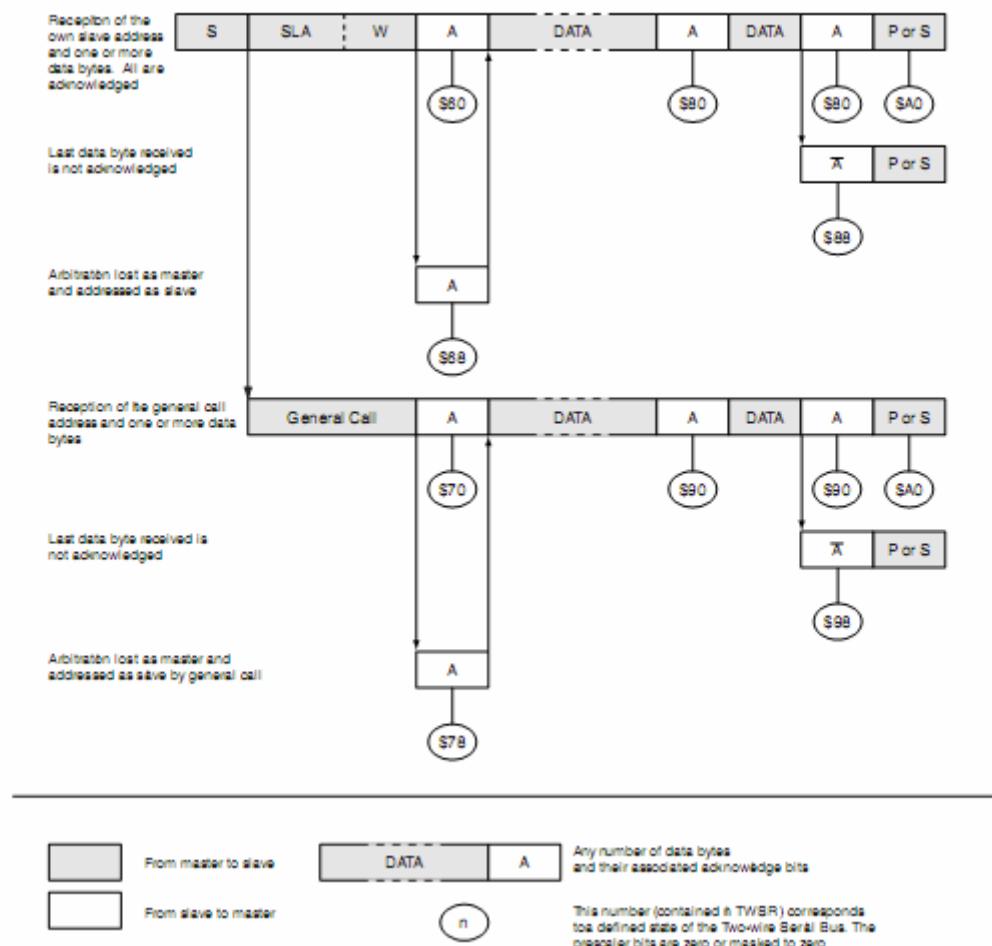
bất cứ thời gian nào bằng việc cài đặt TWEA . Điều này ngụ ý rằng bit TWEA có thể được sử dụng để ngắt tạm thời TWI khỏi bus 2 dây tuần tự

Trong tất cả các chế độ ngủ khác hơn là chế độ Idle , xung nhịp hệ thống tới TWI thì được tắt . Nếu bit TWEA được cài đặt , giao diện có thể vẫn nhận ra được địa chỉ slave của riêng nó hoặc địa chỉ gọi chung bằng cách sử dụng xung nhịp bus 2 dây tuần tự như là một nguồn xung nhịp . Bộ phận sau đó sẽ đánh thức khỏi chế độ sleep và TWI sẽ giữ xung nhịp SCL thấp trong suốt quá trình đánh thức và cho đến khi cờ báo TWINT bị xóa (bằng việc viết nó lên 1) Sự nhận dữ liệu khác sẽ được tiếp tục như bình thường , với các xung nhịp AVR đang chạy như bình thường . Chú ý rằng nếu AVR được cài đặt với một thời gian khởi động dài , đường SCL có thể được giữ ở mức thấp trong một thời gian dài , việc khóa các quá trình chuyển dữ liệu khác

Chú ý rằng thanh ghi dữ liệu giao diện 2 dây tuần tự - TWDR không phản ánh byte cuối cùng được đưa ra trên bus khi đang đánh thức từ các chế độ ngủ .

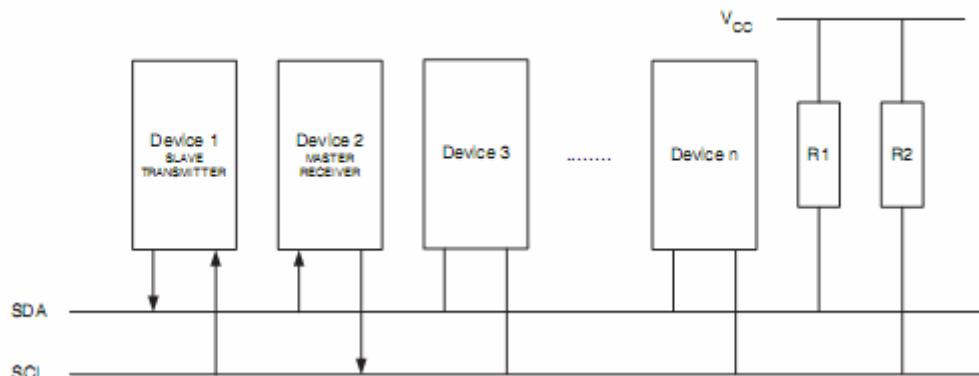
**Table 90. Status Codes for Slave Receiver Mode**

Status Code (TWSR) Prescaler Bits are 0	Status of the Two-wire Serial Bus and Two-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware	
		To TWCR						
		STA	STO	TWINT	TWEA			
\$60	Own SLA+R/W has been received; ACK has been returned;	No TWDR action or No TWDR action	X X	0 0	1 1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$68	Arbitration lost in SLA+R/W as master; own SLA+R/W has been received; ACK has been returned	No TWDR action or No TWDR action	X X	0 0	1 1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$70	General call address has been received; ACK has been returned	No TWDR action or No TWDR action	X X	0 0	1 1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$78	Arbitration lost in SLA+R/W as master; General call address has been received; ACK has been returned	No TWDR action or No TWDR action	X X	0 0	1 1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$80	Previously addressed with own SLA+R/W; data has been received; ACK has been returned	Read data byte or Read data byte	X X	0 0	1 1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$88	Previously addressed with own SLA+R/W; data has been received; NOT ACK has been returned	Read data byte or Read data byte or Read data byte or Read data byte	0 0 1 1	0 0 0 0	1 1 1 1	0 1 0 1	Switched to the not addressed slave mode; no recognition of own SLA or GCA Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1" Switched to the not addressed slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"; a START condition will be transmitted when the bus becomes free	
\$90	Previously addressed with general call; data has been received; ACK has been returned	Read data byte or Read data byte	X X	0 0	1 1	0 1	Data byte will be received and NOT ACK will be returned. Data byte will be received and ACK will be returned.	
\$98	Previously addressed with general call; data has been received; NOT ACK has been returned	Read data byte or Read data byte or Read data byte or Read data byte	0 0 1 1	0 0 0 0	1 1 1 1	0 1 0 1	Switched to the not addressed slave mode; no recognition of own SLA or GCA Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1" Switched to the not addressed slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"; a START condition will be transmitted when the bus becomes free	
\$A0	A STOP condition or repeated START condition has been received while still addressed as slave	No Action	0 0 1 1	0 0 1 1	1 1 0 1	0 1 0 1	Switched to the not addressed slave mode; no recognition of own SLA or GCA Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1" Switched to the not addressed slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"; a START condition will be transmitted when the bus becomes free	

**Figure 101. Formats and States in the Slave Receiver Mode**

## Chế độ bộ chuyển phát Slave ( Slave Transmitter Mode)

Trong chế độ bộ chuyển phát Slave , 1 số của các byte dữ liệu được chuyển phát tới một bộ thu master (xem hình 102 ) . Tất cả các mã trạng thái được nói đến trong phần này giả định rằng các bit bộ đếm gộp trước là 0 hoặc bị che lèn 0 .

**Figure 102. Data Transfer in Slave Transmitter Mode**

Để khởi tạo chế độ bộ chuyển phát Slave , TWAR và TWCR phải được khởi tạo như dưới đây :

TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
value	Device's Own Slave Address							

Hơn 7 bit thì được đánh địa chỉ tới cái mà giao diện 2 dây tuần tự sẽ đáp ứng khi được đánh địa chỉ bởi master . Nếu như LSB được cài đặt , TWI sẽ đáp ứng để đánh địa chỉ gọi chung (\$00) , nói cách khác nó sẽ bỏ qua địa chỉ gọi chung

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	0	1	0	0	0	1	0	X

TWEN phải được ghi là 1 để kích hoạt TWI . bit TWEA phải được ghi là 1 để kích hoạt sự thửa nhận của địa chỉ slave của riêng thiết bị hoặc địa chỉ gọi chung . TWSTA và TWSTO phải được ghi là 0 .

Khi TWAR và TWCR vừa được khởi tạo , TWI đợi cho đến khi nó được đánh địa chỉ bởi slave riêng của nó (hoặc địa chỉ lệnh gọi chung nếu được kích hoạt ) được theo bởi bit định hướng dữ liệu . Nếu bit định hướng là 1 (đọc ) , TWI sẽ hoạt động trong chế độ ST , nói cách khác chế độ SR được truy nhập . Sau khi địa chỉ slave riêng của nó và bit ghi vừa được nhận , cờ báo TWINT được cài đặt và một mã trạng thái có hiệu lực có thể đọc từ TWSR . Mã trạng thái được sử dụng để xác định hành động thích đáng của phần mềm . Hành động thích đáng của phần mềm tạo ra trong mỗi mã trạng thái được nêu chi tiết trong bảng 91 . Chế độ bộ nhận Slave cũng có thể được truy nhập nếu như quá trình giám định bị mất trong khi TWI trong chế độ master (xem các trạng thái \$B0)

Nếu bit TWEA được viết là 0 trong suốt 1 quá trình chuyển phát , TWI sẽ chuyển byte dữ liệu cuối cùng của quá trình chuyển phát . Trạng thái \$C0 và trạng thái \$C8 sẽ được truy nhập , phụ thuộc vào bộ nhận master chuyển phát 1 NACK và ACK sau khi byte kết thúc . TWI được chuyển mạch tới chế độ slave không được đánh địa chỉ , và sẽ bỏ qua master nếu nó tiếp tục truyền phát . Vì vậy bộ thu nhận master nhận tất cả “1” như là dữ liệu nối tiếp . Trạng thái \$C8 được truy nhập nếu các byte dữ liệu thêm vào sự yêu cầu master (bằng việc truyền bit ACK), mặc dù slave vừa truyền byte cuối cùng (TWEA là 0 và đang mong đợi NACK từ master)

Trong khi TWEA là 0 , TWI không đáp ứng tới địa chỉ slave của riêng nó . Tuy nhiên , bus 2 dây nối tiếp vẫn được giám sát và sự nhận diện địa chỉ có thể khôi phục tại bất cứ thời gian nào bằng việc cài đặt TWEA . Điều này tức là TWEA có thể sử dụng để ngắt tạm thời TWI khỏi bus 2 dây tuần tự .

Trong tất cả các chế độ ngủ khác hơn là chế độ Idle , xung nhịp hệ thống tới TWI thì được tắt . Nếu bit TWEA được cài đặt , giao diện có thể vẫn nhận ra được địa chỉ slave của riêng nó hoặc địa chỉ gọi chung bằng cách sử dụng xung nhịp bus 2 dây tuần tự như là một nguồn xung nhịp . Bộ phận sau đó sẽ đánh thức khỏi chế độ sleep và TWI sẽ giữ xung nhịp SCL thấp trong suốt quá trình đánh thức và cho đến khi cờ báo TWINT bị xóa (bằng việc viết nó lên 1) Sự nhận dữ liệu khác sẽ được tiếp tục như bình thường , với các xung nhịp AVR đang chạy như bình thường . Chú ý rằng nếu AVR được cài đặt với một thời gian khởi động dài , đường SCL có thể được giữ ở mức thấp trong một thời gian dài , việc khóa các quá trình chuyển dữ liệu khác

Trong tất cả các chế độ ngủ khác hơn là chế độ Idle , xung nhịp hệ thống tới TWI thì được tắt . Nếu bit TWEA được cài đặt , giao diện có thể vẫn nhận ra được địa chỉ

slave của riêng nó hoặc địa chỉ gọi chung bằng cách sử dụng xung nhịp bus 2 dây tuần tự như là một nguồn xung nhịp. Bộ phận sau đó sẽ đánh thức khỏi chế độ sleep và TWI sẽ giữ xung nhịp SCL thấp trong suốt quá trình đánh thức và cho đến khi cờ báo TWINT bị xóa (bằng việc viết nó lên 1) Sự nhận dữ liệu khác sẽ được tiếp tục như bình thường, với các xung nhịp AVR đang chạy như bình thường.

Chú ý rằng nếu AVR được cài đặt với một thời gian khởi động dài, đường SCL có thể được giữ ở mức thấp trong một thời gian dài, việc khóa các quá trình chuyển dữ liệu khác

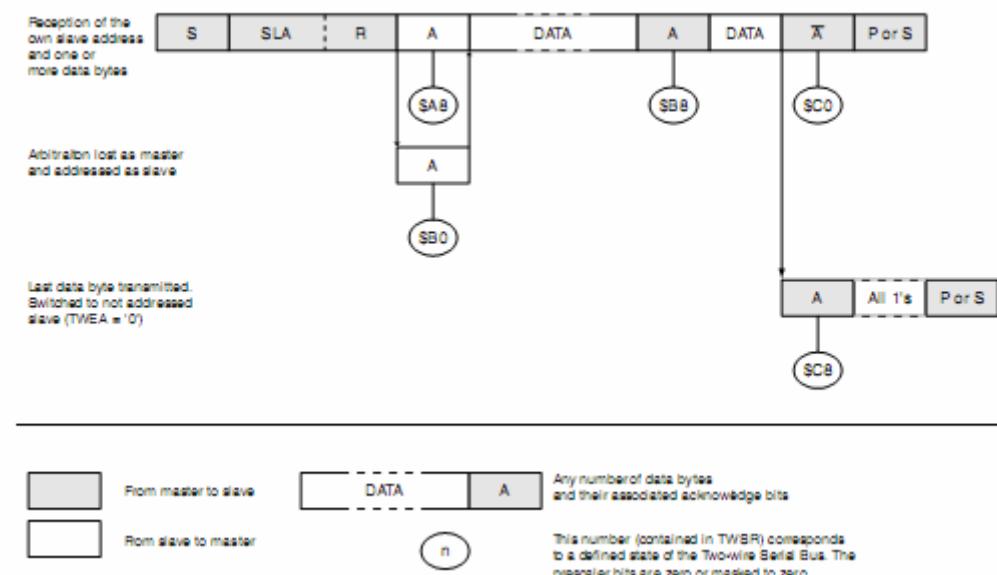
Table 91. Status Codes for Slave Transmitter Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the Two-wire Serial Bus and Two-wire Serial Interface Hardware	Application Software Response				Next Action Taken by TWI Hardware
		To/TWDR		STA STO TWINT TWEA		

Table 91. Status Codes for Slave Transmitter Mode

SA8	Own SLA+R has been received; ACK has been returned	Load data byte or	X	0	1	0	Last data byte will be transmitted and NOT ACK should be received
		Load data byte	X	0	1	1	Data byte will be transmitted and ACK should be received
SB0	Arbitration lost in SLA+R/W as master; own SLA+R has been received; ACK has been returned	Load data byte or	X	0	1	0	Last data byte will be transmitted and NOT ACK should be received
		Load data byte	X	0	1	1	Data byte will be transmitted and ACK should be received
SB8	Data byte in TWDR has been transmitted; ACK has been received	Load data byte or	X	0	1	0	Last data byte will be transmitted and NOT ACK should be received
		Load data byte	X	0	1	1	Data byte will be transmitted and ACK should be received
SC0	Data byte in TWDR has been transmitted; NOT ACK has been received	No TWDR action or	0	0	1	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA
		No TWDR action or	0	0	1	1	Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"
		No TWDR action or	1	0	1	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
		No TWDR action	1	0	1	1	Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"; a START condition will be transmitted when the bus becomes free
SC8	Last data byte in TWDR has been transmitted (TWEA = "0"); ACK has been received	No TWDR action or	0	0	1	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA
		No TWDR action or	0	0	1	1	Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"
		No TWDR action or	1	0	1	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
		No TWDR action	1	0	1	1	Switched to the not addressed slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = "1"; a START condition will be transmitted when the bus becomes free

Figure 103. Formats and States in the Slave Transmitter Mode



## Các trạng thái hỗn hợp

Trạng thái \$F8 hiển thị rằng không có thông tin xác đáng nào là khả dụng bởi vì cờ báo TWINT không được cài đặt. Điều này xuất hiện giữa các trạng thái khác, và khi TWI không được chứa trong một quá trình chuyển phát nối tiếp.

Trạng thái \$00 hiển thị rằng 1 bus lỗi vừa xuất hiện trong suốt quá trình chuyển dữ liệu của bus 2 dây tuần tự. 1 bus lỗi xuất hiện khi khi 1 điều kiện START hoặc STOP xuất hiện tại một vị trí không hợp lệ trong dạng khung truyền. Các mẫu của các vị trí không hợp lệ trong suốt quá trình chuyển dữ liệu nối tiếp của một byte địa chỉ, một byte dữ liệu, hoặc một bit nhận biết. Khi một bus lỗi xuất hiện, TWINT được cài đặt. Để khôi phục từ một bus lỗi, cờ TWSTO phải được cài đặt và TWINT phải bị xóa bằng việc viết mức logic 1 lên nó. Điều này gây ra TWI truy nhập vào chế độ Slave không địa chỉ để xóa cờ TWSTO (không có bit nào khác trong TWCR bị ảnh hưởng). Các đường SDA và SCL được giải phóng, và không có điều kiện STOP nào được chuyển đi.

Table 92. Miscellaneous States

Status Code (TWCSR) Prescaler Bits are 0	Status of the Two-wire Serial Bus and Two-wire Serial Interface Hardware	Application Software Response				Next Action Taken by TWI Hardware	
		To TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
\$F8	No relevant state information available; TWINT = "0"	No TWDR action	No TWCR action			Wait or proceed current transfer	
\$00	Bus error due to an illegal START or STOP condition	No TWDR action	0	1	1	X	Only the internal hardware is affected, no STOP condition is sent on the bus. In all cases, the bus is released and TWSTO is cleared.

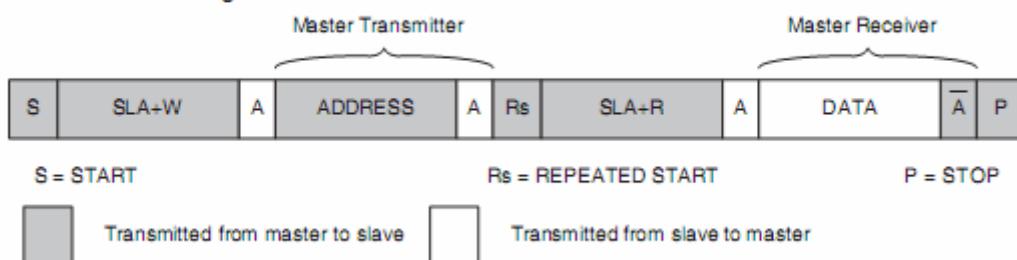
## Việc kết nối nhiều chế độ TWI nối tiếp

Trong một vài trường hợp , các chế độ TWI phải được kết nối với nhau để hoàn thành các hành động thêm . Coi như việc đọc dữ liệu từ 1 nhiều bộ nhớ EEPROM . Thông thường , như là một quá trình chuyển phát chứa các bước dưới đây :

1. Sự chuyển phát phải được khởi tạo
2. EEPROM phải được hướng dẫn vùng địa chỉ mà nên đọc
3. quá trình đọc phải được tiến hành
4. quá trình chuyển phát phải được kết thúc

chú ý rằng dữ liệu được chuyển từ master đến slave và vice versa . Master phải hướng dẫn slave vùng địa chỉ mà nó muốn đọc , sự cần thiết phải sử dụng của chế độ MT . Như một sự xảy ra sau , dữ liệu phải được đọc từ slave , và tiến hành sử dụng chế độ MR . Vì vậy , hướng chuyển dữ liệu phải được thay đổi . Master phải giữ sự điều khiển bus trong suốt những bước tiếp theo , và các bước này nên được tiếp tục như một quá trình điều khiển nguyên tử . Nếu nguyên tắc này bị vi phạm trong một hệ thống nhiều master, các master khác sau đó có thể đánh dấu dữ liệu trong EEPROM giữa bước 2 và bước 3 , và master sẽ đọc vùng địa chỉ dữ liệu sai . Như là sự thay đổi trong hướng chuyển dữ liệu thì được hoàn thành bằng việc chuyển 1 điều kiện REPEATED START giữa sự chuyển phát của byte địa chỉ và sự tiếp nhận dữ liệu . Sau một REPEATED START, master giữ quyền sở hữu của bus . Hình dưới đây chỉ ra dòng trong việc chuyển phát này

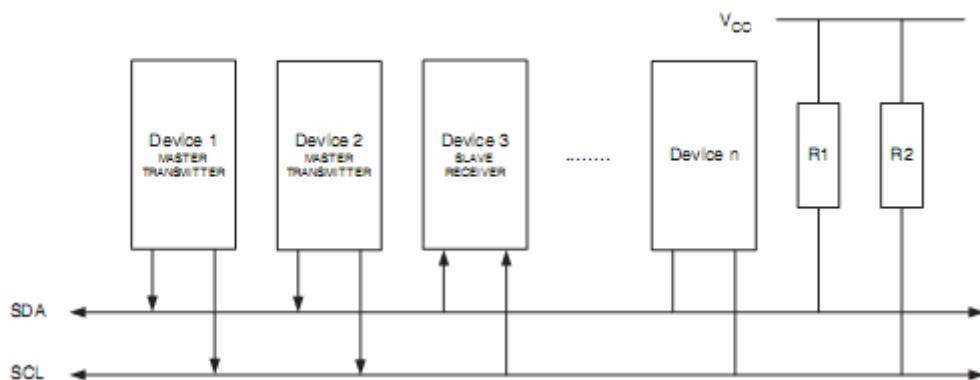
**Figure 104. Combining Several TWI Modes to Access a Serial EEPROM**



## Hệ thống nhiều master và sự kiểm định

Trong một hệ thống nhiều master được kết nối đến các bus giống nhau , các quá trình truyền có thể khởi tạo một cách đồng thời bằng 1 hoặc nhiều trong số chúng .TWI tiêu chuẩn đảm bảo rằng như các trường hợp được điều khiển trong 1 cách mà 1 trong số nhiều master sẽ cho phép để tiến hành với sự chuyển phát , và không có dữ liệu nào bị mất trong tiến trình đó . Một ví dụ của một sự tình huống kiểm định được miêu tả bên dưới , nơi mà 2 master đang cố gắng chuyển dữ liệu đến một slave

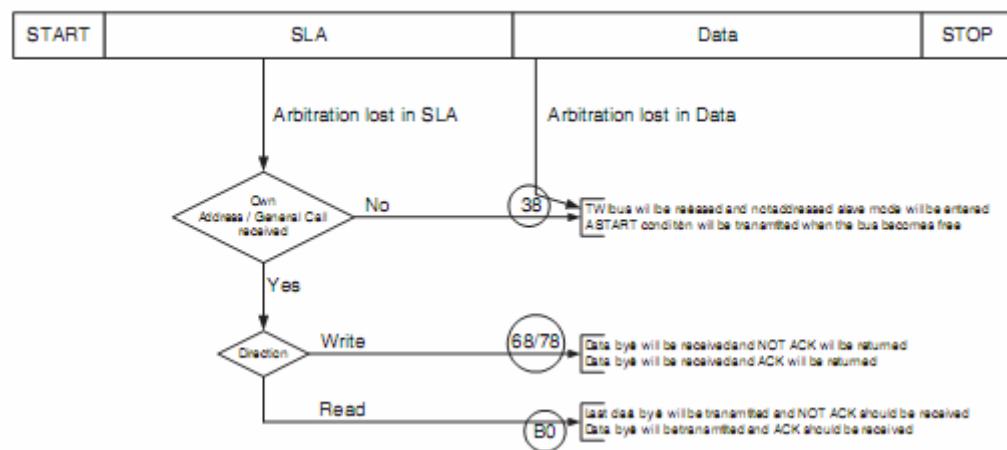
Figure 105. An Arbitration Example



Nhiều tình huống khác nhau có thể phát sinh trong quá trình kiểm định , như là được miêu tả dưới đây

- 2 hoặc nhiều master được đang sử dụng truyền thông riêng biệt với các slave giống nhau . Trong trường hợp này , hoặc là slave hoặc là bất cứ master nào sẽ biết về sự tranh chấp bộ nhớ
- 2 hoặc nhiều master đang truy nhập vào slave giống nhau với dữ liệu khác nhau hoặc bit định hướng . Trong trường hợp này , sự kiểm định sẽ xuất hiện , hoặc là trong bit đọc/viết hoặc là trong các bit dữ liệu . Các master đang cố gắng để xuất ra 1 trên SDA trong khi các đầu ra master khác là 0 sẽ làm mất sự kiểm định . Sự mất mát của master sẽ chuyển mạch tới chế độ slave không địa chỉ hoặc đợi cho đến khi bus là tự do và truyền một điều kiện START mới , phụ thuộc vào hành động của phần mềm ứng dụng .
- 2 hoặc nhiều master đang truy nhập vào các slave giống nhau . Trong trường hợp này , quá trình kiểm định sẽ xuất hiện trong các bit SLA . Các master đang cố gắng xuất ra một tín hiệu 1 trên SDA trong khi các đầu ra master khác là 0 sẽ làm mất dữ liệu .Sự mất kiểm định của các master trong SLA sẽ chuyên đến chế độ slave để kiểm tra nếu chúng đang được đánh địa chỉ bởi master chiến thắng. Nếu được đánh địa chỉ , chúng sẽ chuyển tới chế độ SR hoặc ST , phụ thuộc vào giá trị của bit READ/WRITE . Nếu chúng đang được đánh địa chỉ , chúng sẽ chuyển tới chế độ Slave không đánh địa chỉ hoặc đợi cho đến khi bus rỗi và chuyển phát một điều kiện START mới, phụ thuộc vào hành động của phần mềm ứng dụng .

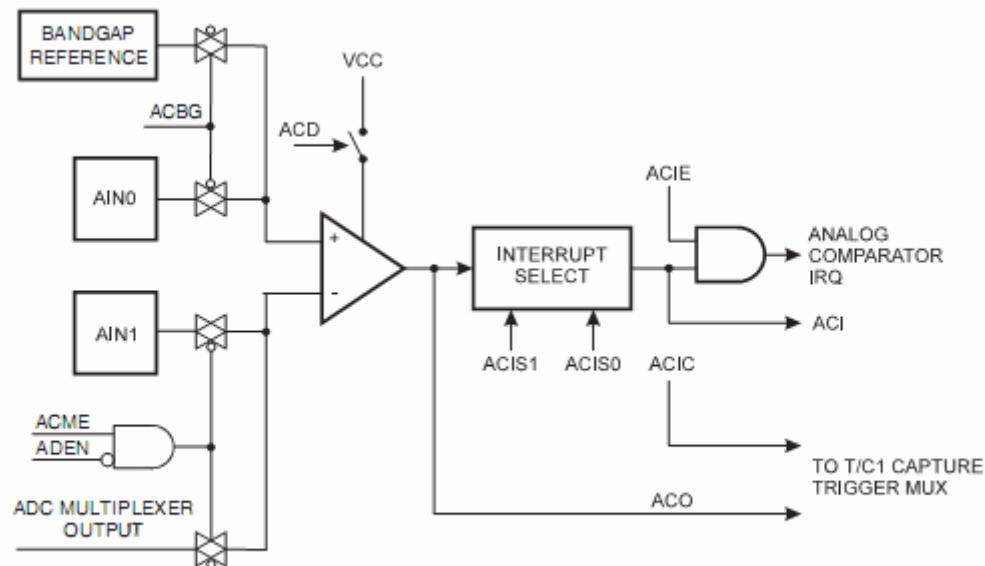
Điều này được mô tả chi tiết trong hình 106. giá trị trạng thái có thể được đưa ra trong vòng tròn

**Figure 106.** Possible Status Codes Caused by Arbitration

## XX . Analog comparator

Bộ so sánh tương tự so sánh các giá trị của đầu vào trên chân dương AIN0 và chân âm AIN1 . Khi điện áp trên chân dương AIN0 cao hơn điện áp trên chân âm AIN1 , đầu ra của bộ so sánh tương tự ACO được cài đặt . Đầu ra của bộ so sánh có thể được cài đặt để khởi động chức năng truy bắt tín hiệu đầu vào của Timer/Counter 1 .Thêm vào đó , bộ so sánh có thể khởi động 1 ngắt riêng biệt , dành riêng cho bộ so sánh tương tự . Người sử dụng có thể lựa chọn khởi động ngắt của bộ so sánh đầu ra tăng, giảm hoặc di chuyển. Một sơ đồ khối của bộ so sánh và các khối logic xung quanh nó được chỉ ra ở hình 107 .

**Figure 107.** Analog Comparator Block Diagram



Notes:

1. See Table 94 on page 229.
2. Refer to Figure 1 on page 2 and Table 39 on page 81 for Analog Comparator pin placement.

## Thanh ghi IO chức năng đặc biệt\_SFIOR

Bit3\_ACME : Kích hoạt bộ dòn kênh của bộ so sánh tương tự

Khi bít này được viết là 1 và bộ chuyển đổi ADC được tắt (Bít ADEN trong thanh ghi ADCSRA là 0 ) , bộ dòn kênh ADC lựa chọn đầu vào âm lên bộ so sánh tương tự . Khi bít này được viết là 0 , AIN1 được đặt lên đầu vào âm của bộ so sánh tương tự để biết thêm chi tiết của bít này xem trang 228

## Thanh ghi trạng thái và điều khiển bộ so sánh tương tự ACSR

Bít 7\_ACD : Vô hiệu hóa bộ so sánh tương tự

Khi bít này được viết là 1 , nguồn cung cấp tới bộ so sánh tương tự bị ngắt . Bít này có thể được cài đặt tại bất cứ thời điểm nào để tắt bộ so sánh tương tự. Điều này sẽ làm giảm tốn hao nguồn trong các chế độ Active và Idle . Khi thay đổi bít ACD , các ngắt của bộ so sánh tương tự phải được vô hiệu hóa bằng việc xóa bít ACIE trong thanh ghi ACSR . Nói cách khác, 1 ngắt có thể xuất hiện khi bít này bị thay đổi

Bít 6\_ACBG : Lựa chọn khe song (bandgap) của bộ so sánh tương tự

Khi bít này được cài đặt , một bandgap ổn định điện áp tham chiếu thay thế đầu vào dương đến bộ so sánh tương tự . Khi bít này bị xóa , AIN0 được đặt lên đầu vào dương của bộ so sánh tương tự . Xem thêm phần Internal Voltage Reference trang 54

Bít 5\_ACO : Đầu vào của bộ so sánh tương tự

Đầu vào của bộ so sánh tương tự được đồng bộ hóa và sau đó được nối trực tiếp tới ACO , sự đồng bộ hóa này gây ra 1 trễ trong 1 hoặc 2 chu kỳ xung nhịp

Bít 4\_ACI : Cờ báo ngắt bộ so sánh tương tự

Bít này được cài đặt bằng phần cứng khi mà 1 biến cố ở đầu ra của bộ so sánh khởi động 1 chế độ ngắt được xác định bởi bít ACIS1 và ACIS0 . Chương trình con phục vụ ngắt bộ so sánh tương tự được thực thi nếu bít ACIE được cài đặt và bít I trong thanh ghi SGDN được cài đặt . Bít ACI bị xóa bằng phần cứng khi mà việc thực thi các vector điều khiển ngắt tương ứng được hoàn thành . Như 1 sự lựa chọn , ACI được xóa bằng cách viết mức logic 1 lên cờ ngắt

Bít 3\_ACIE : Kích hoạt ngắt bộ so sánh tương tự

Khi mà bít ACIE được viết là 1 và bít I trong thanh ghi trạng thái được cài đặt , ngắt của bộ so sánh tương tự được kích hoạt . Khi được viết là 0 thì ngắt được vô hiệu hóa

Bít 2\_ACIC : Kích hoạt truy bắt tín hiệu đầu vào của bộ so sánh tương tự

Khi được viết là 1 bít này kích hoạt chức năng truy bắt tín hiệu đầu vào trong Timer/Counter 1 để khởi động bằng bộ so sánh tương tự . Đầu ra bộ so sánh trong trường hợp này được nối trực tiếp đến cổng logic front-end của bộ truy bắt tín hiệu đầu vào làm cho bộ so sánh tận dụng các khóa cắt nhiễu và tính năng lựa chọn sườn của ngắt truy bắt tín hiệu đầu vào Timer/Counter 1 . Khi được viết là 0 , không có kết nối nào giữa bộ so sánh tương tự và đầu ra của khối chức năng truy bắt tín hiệu đầu vào . Để làm cho bộ so sánh khởi động ngắt truy bắt tín hiệu đầu vào timer/counter 1 . Bít TICIE1 trong thanh ghi che ngắt Timer (TIMSK) phải được cài đặt

Bít 1,0 : Lựa chọn chế độ ngắt so sánh tương tự

Các bít này xác định cái mà các biến cố của bộ so sánh tương tự khởi động các ngắt của bộ so sánh tương tự . Việc cài đặt khác nhau được minh họa ở bảng 93

**Table 93. ACIS1/ACIS0 Settings**

<b>ACIS1</b>	<b>ACIS0</b>	<b>Interrupt Mode</b>
0	0	Comparator Interrupt on Output Toggle
0	1	Reserved
1	0	Comparator Interrupt on Falling Output Edge
1	1	Comparator Interrupt on Rising Output Edge

Khi thay đổi các bít ACIS1/ACIS0 các ngắt của bộ so sánh tương tự phải được vô hiệu hóa bằng việc xóa các bít kích hoạt ngắt của nó trong thanh ghi ACSR , nói cách khác, 1 ngắt có thể xuất hiện khi mà các bít này có thể bị thay đổi

### Đầu vào dòn kênh của bộ so sánh tương tự

Có thể lựa chọn bất cứ chân nào trong các chân ADC7..0 để thay thế đầu vào âm đến bộ so sánh tương tự . Bộ dòn kênh ADC được sử dụng để lựa chọn đầu vào này kéo theo việc ADC phải được tắt để dùng tính năng này . Nếu như bít kích hoạt bộ dòn kênh của bộ so sánh tương tự (ACME trong SFIOR) được cài đặt và bộ chuyển đổi ADC được tắt (Bít ADEN trong thanh ghi ADCSRA là 0 ) , các bít MUX2..0 trong thanh ghi ADMUX lựa chọn các chân vào này để thay thế chân vào âm đến bộ so sánh tương tự , được chỉ ra trong bảng 94 . Nếu ACME bị xóa hoặc ADEN được cài đặt , bít AIN1 được đặt lên đầu vào âm tới bộ so sánh tương tự

**Table 94. Analog Comparator Multiplexed Input**

<b>ACME</b>	<b>ADEN</b>	<b>MUX2..0</b>	<b>Analog Comparator Negative Input</b>
0	x	xxx	AIN1
1	1	xxx	AIN1
1	0	000	ADC0
1	0	001	ADC1
1	0	010	ADC2
1	0	011	ADC3
1	0	100	ADC4
1	0	101	ADC5
1	0	110	ADC6
1	0	111	ADC7

## XXI . Bộ chuyển đổi tương tự sang số (Analog to Digital Converter )

Các đặc điểm :

- Độ chính xác 10 bít
- 0.5 LSB Integral Non-Linearity
- Khoảng tuyết đối  $\pm 2$  LSB
- Thời gian chuyển đổi từ 13 - 260 $\mu$ s
- Nâng lên 76,9 kSPS ( Nâng lên 15 kSPS ) ở độ chính xác cực đại
- 8 kênh multiplexed single ended input
- 7 kênh đầu vào riêng biệt
- 2 kênh đầu vào riêng biệt với các độ khuyêch đại
- Sự điều chỉnh dy chuyển , lựa chọn cho kết quả đọc bên ngoài ADC
- 0\_VCC dải điện áp đầu vào ADC
- Có thể lựa chọn 2,56V điện áp tham chiếu ADC
- Chế độ chuyển đổi đơn hoặc chạy tự do
- Ngắt trên hoàn thành quá trình chuyển đổi ADC
- Khóa cắt nhiễu chế độ ngủ (Sleep mode )

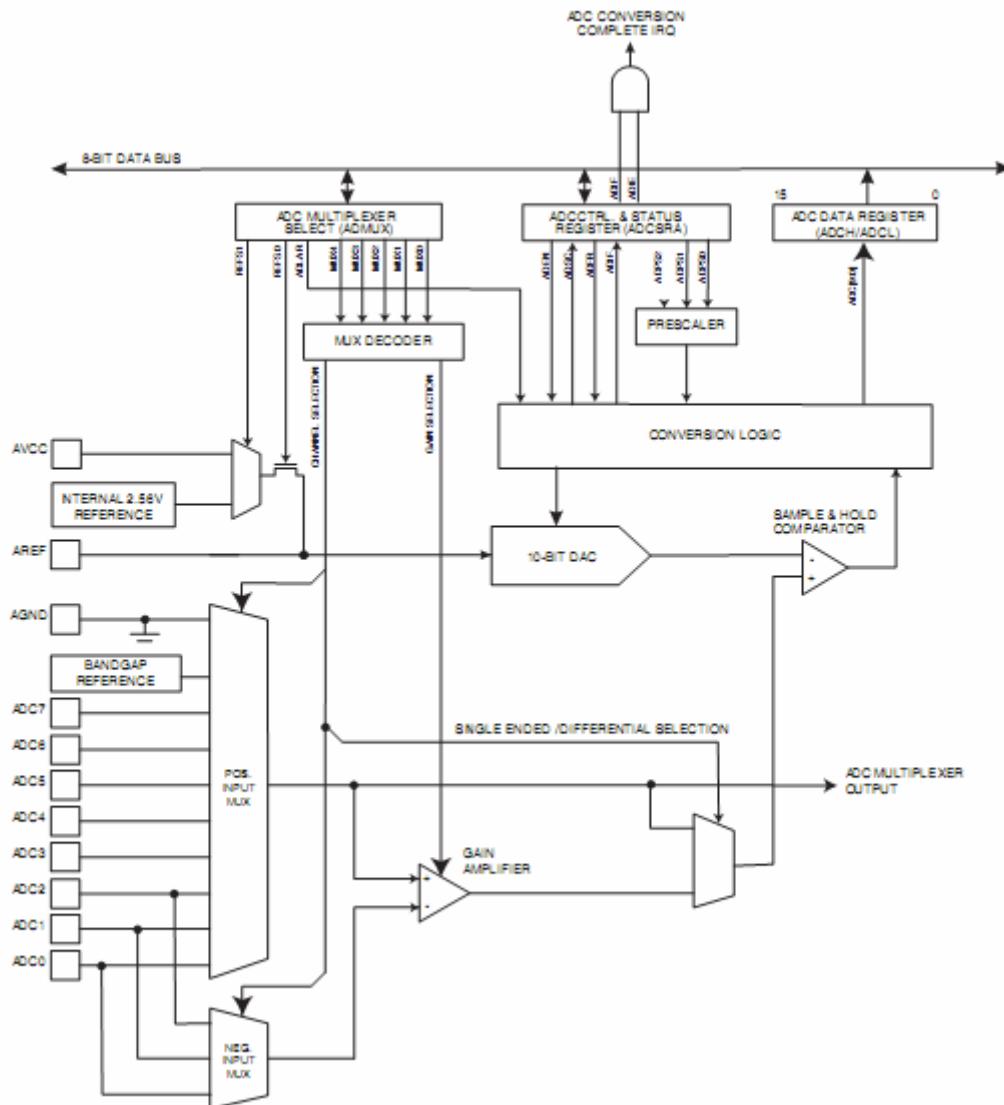
Các tính năng của Atmega128 có 1 bộ ADC sấp xỉ liên tiếp 10 bít . Bộ ADC được kết nối đến 1 bộ dồn kênh tương tự 8 kênh, cái mà cho phép 8 đầu vào điện áp Singel ended . Các đầu vào điện áp single ended được đặt là 0 (chân GND)

Thiết bị này cũng hỗ trợ 16 kết nối đầu vào điện áp riêng biệt , 2 cặp đầu vào riêng biệt (ADC1 , ADC0 và ADC3, ADC2) được trang bị với 1 công khuyech đại có thể lập trình , việc cung cấp các bậc khuyech đại là 0dB (1x) , 20dB(10x) , 46dB(200x) trên các cổng điện áp riêng biệt trước khi có quá trình chuyển đổi A/D . 7 kênh đầu vào tương tự riêng biệt chia sẻ 1 đầu cuối âm chung (ADC1) , trong khi bắt cứ các đầu vào ADC khác đều có thể được lựa chọn như là 1 đầu cuối đầu vào dương . Nếu độ khuyech đại 1x hoặc 10x được sử dụng , độ chính xác 8 bít có thể được chờ đợi . Nếu độ khuyech đại 200x được sử dụng , độ chính xác 7bit có thể được chờ đợi .

Bộ ADC bao gồm 1 mạch lấy mẫu và mạch giữ mức cái mà đảm bảo rằng điện áp đầu vào đến ADC được giữ ở 1 mức không đổi trong suốt quá trình chuyển đổi . Sơ đồ khối của 1 bộ ADC được chỉ ra trong hình 108

Bộ ADC có 1 chân điện áp nguồn cấp tương tự riêng biệt , AVCC . AVCC phải khác biệt hơn 0,3V từ chân VCC . Xem phần khóa cắt nhiễu ADC trên trang 236 để biết cách kết nối các chân này . Các điện áp tham chiếu bên trong của điện áp thông thường 2,56V hoặc AVCC được cung cấp trên chip . Điện áp tham chiếu có thể tách rời tại chân AREF bằng 1 tụ để có được các đường đặc tính nhiễu tốt hơn

**Figure 108.** Analog to Digital Converter Block Schematic



## **Hoạt động**

Bộ ADC chuyển đổi 1 điện áp đầu vào tương tự thành 1 giá trị số 10 bít thông qua các phép tính xấp xỉ liên tiếp . Giá trị cực tiểu đưa ra ở chân GND và giá trị cực đại của điện áp trên chân AREF tối thiểu là 1 LSB . Sự lựa chọn giữa AVCC hoặc 1 điện áp tham chiếu bên trong 2,56 V có thể được kết nối đến chân AREF bằng việc viết lên các bít RESFn ở trong thanh ghi ADMUX . Điện áp tham chiếu bên trong có thể được tách rời bằng 1 tụ bên ngoài ở chân AREF để cải thiện các nhiễu .

Kênh đầu vào tương tự và các hệ số khuyech đại riêng biệt bằng việc viết lên các bít MUX trong thanh ghi ADMUX bất cứ chân nào trong số các chân đầu vào ADC như là chân GND và điện áp tham chiếu bandgap ổn định , đều có thể được lựa chọn như là các đầu vào Singel ended lên các chân ADC , 1 sự lựa chọn của các chân đầu vào ADC có thể được lựa chọn như là chân âm và dương lên cách bộ khuyech đại riêng biệt

Nếu các kênh riêng biệt được lựa chọn, cổng khuỷu chệch đại riêng biệt sẽ khuỷu chệch đại sự chênh lệch điện áp giữa các cặp kênh đầu vào được lựa chọn bằng việc lựa chọn

hệ số khuyech đại . giá trị được khuyech đại sau đó được trở thành đầu vào tương tự trong ADC . Nếu các kênh single ended được sử dụng thì các bộ khuyech đại được chuyển qua cùng nhau .

Bộ ADC được kích hoạt bằng cách cài đặt các bít kích hoạt ADC (ADEN trong ADCSRA). Điện áp tham chiếu và sự lựa chọn kênh đầu vào sẽ không được gây ảnh hưởng vào bên trong cho đến khi bít ADEN được cài đặt bộ ADC thì không tiêu hao nguồn khi mà bít ADEN bị xóa vì vậy nó được khuyến cáo để ngắt các bộ ADC trước khi truy nhập vào chế bộ tiết kiệm điện trong các chế độ ngủ

Bộ ADC phát ra 1 kết quả 10 bít , cái mà được đưa ra trong thanh ghi dữ liệu ADC (ADCH và ADCL ) . Như mặc định , kết quả này được đưa ra 1 sự điều chỉnh đúng nhưng nó có thể lựa chọn để được đưa ra sự điều chỉnh di chuyển bằng việc cài đặt bít ADLAR trong thanh ghi ADMUX .

Nếu như kết quả được điều chỉnh di chuyển và không cần độ chính xác lớn hơn 8 bít . Nó sẽ được tiến hành để đọc trong ADCH , nói cách khác ADCL có thể được đọc đầu tiên sau đó ADCH được đọc , để đảm bảo rằng các thành phần của các thanh ghi dữ liệu thuộc về các quá trình chuyển đổi giống nhau . Mỗi lần ADCL được đọc thì bộ ADC truy cập vào các thanh ghi dữ liệu bị khóa . Điều này có nghĩa là nếu ADCL vừa được đọc và 1 quá trình chuyển đổi hoàn thành trước khi ADCH được đọc thì thanh ghi được cập nhật dữ liệu và kết quả từ quá trình chuyển đổi đó bị mất . Khi mà ADCH được đọc ADC truy cập đến các thanh ghi ADCL ADCH để kích hoạt lại .

Bộ ADC có các ngắt riêng của nó , cái mà có thể được khởi động khi mà có 1 sự chuyển đổi hoàn thành . Khi bộ ADC truy nhập vào các thanh ghi dữ liệu bị ngăn cấm giữa quá trình đọc của ADCH và ADCL , các ngắt sẽ khởi động dù nếu kết quả bị mất

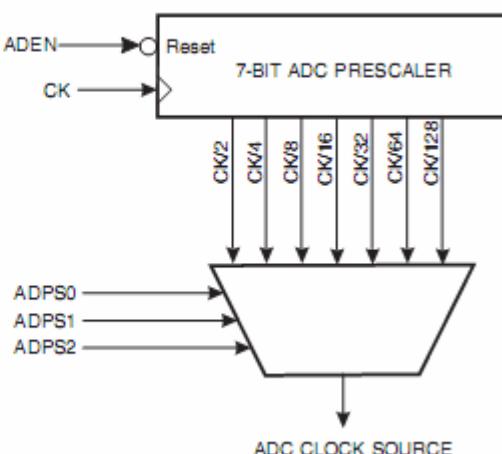
## **Khởi động 1 quá trình chuyển đổi**

1 quá trình chuyển đổi đơn được khởi động bằng việc viết mức logic 1 lên bít khởi động chuyển đổi ADC (ADSC ) . Bít này ở mức cao chỉ cần quá trình chuyển đổi đang tiến hành và sẽ bị xóa bằng phần cứng khi mà quá trình chuyển đổi hoàn thành . Nếu như 1 kênh dữ liệu khác được lựa chọn trong khi 1 quá trình chuyển đổi đang tiến hành, bộ ADC sẽ kết thúc quá trình chuyển đổi hiện tại trước khi sử lý thay đổi kênh .

Trong chế độ chạy tự do, bộ ADC lấy mẫu và cập nhật thanh ghi dữ liệu ADC . Chế độ chạy tự do được lựa chọn bằng việc viết bít ADFR trong thanh ghi ADCSRA là 1 . Quá trình chuyển đổi đầu tiên phải được bắt đầu bằng việc viết mức logic 1 lên bít ADSC trong thanh ghi ADCSRA . Trong chế độ này , bộ ADC sẽ tiến hành các quá trình chuyển đổi liên tiếp phụ thuộc vào trạng thái của cờ ngắt ADC (ADIF ) bị xóa hay không .

## Sự đếm gộp trước và giản đồ thời gian của quá trình chuyển đổi

Figure 109. ADC Prescaler



Theo mặc định các mạch tính sáp xỉ liên tiếp cần thiết 1 tần số xung nhịp đầu vào trong khoảng 50 KHz – 200 KHz để đạt độ chính xác cực đại . Nếu 1 độ chính xác thấp hơn 10 bít là cần thiết , tần số xung nhịp đầu vào lên ADC có thể cao hơn 200KHz để có 1 tốc bộ lấy mẫu cao hơn

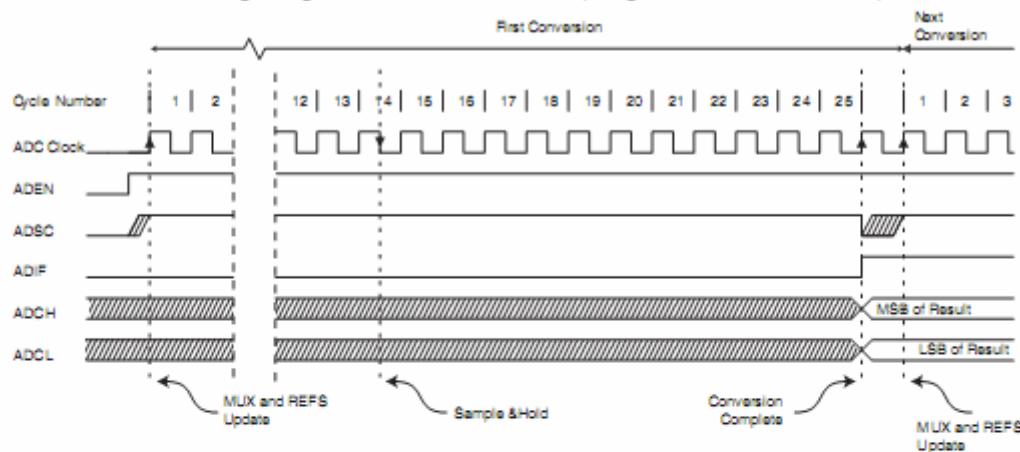
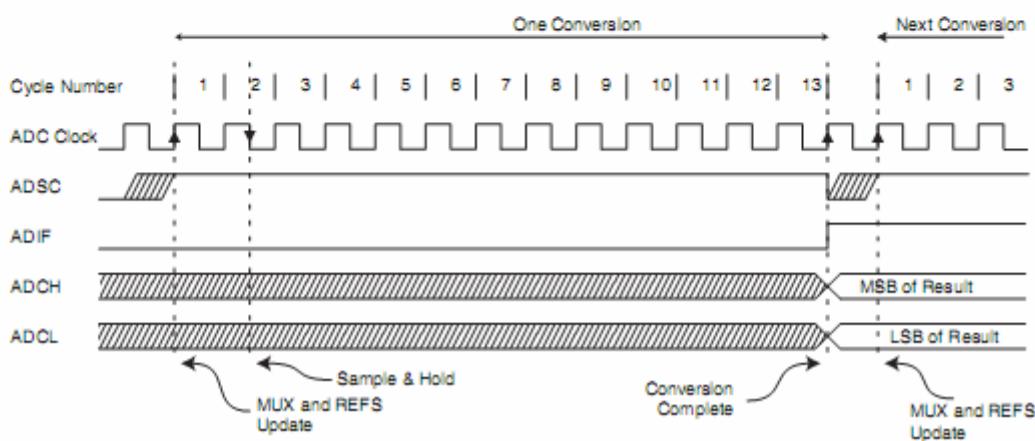
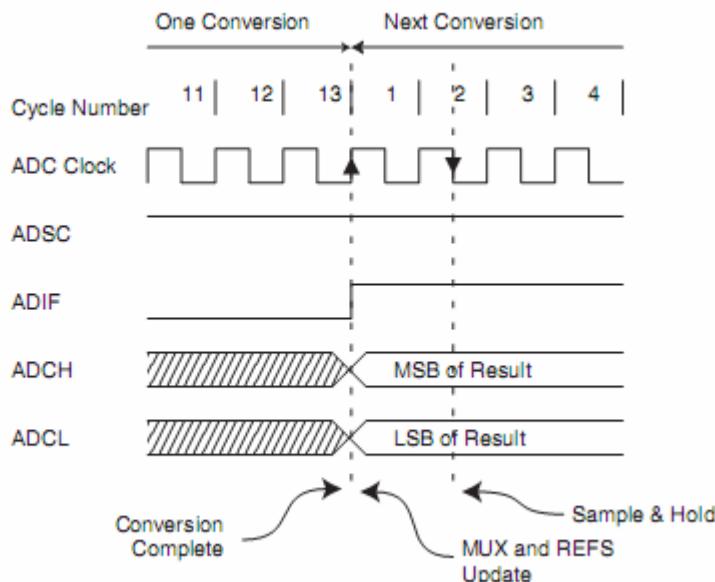
Modul ADC bao gồm 1 bộ đếm gộp trước , cái mà phát ra 1 tần số xung nhịp ADC có thể chấp nhận được từ bất cứ 1 tần số CPU nào dưới 100Khz . Sự đếm gộp trước được cài đặt bằng các bít ADPS trong thanh ghi ADCSRA . Bộ đếm gộp trước bắt đầu đếm từ lúc mà bộ ADC bị cắt bằng việc cài đặt bít ADEN trong thanh ghi ADCSRA . Bộ đếm gộp trước vẫn giữ đang chạy chỉ cần bít ADEN được cài đặt và liên tục khởi động lại khi ADEN ở mức thấp

Khi khởi tạo 1 chuyển đổi single ended bằng việc cài đặt bít ADSC trong thanh ghi ADCSRA , quá trình chuyển đổi bắt đầu từ sùn lên kế tiếp của chu kì xung nhịp ADC . Xem phần differential gain chanel trên trang 235 để biết thêm chi tiết về giản đồ thời gian của quá trình chuyển đổi riêng biệt

Một quá trình chuyển đổi thông thường tạo ra 13 chu kì xung nhịp ADC . Chu kì xung nhịp đầu tiên sau khi ADC được đóng (ADEN trong ADCSRA được cài đặt ) tạo ra 25 xung nhịp ADC theo thứ tự khởi tạo mạch tương tự .

Thực tế quá trình lấy mẫu và giữ mức xảy ra trong 1,5 chu kì xung nhịp ADC sau khi bắt đầu 1 quá trình chuyển đổi thông thường và 13,5 xung nhịp ADC sau khi bắt đầu 1 quá trình chuyển đổi đầu tiên . Khi 1 quá trình chuyển đổi được hoàn thành , kết quả được viết lên các thanh ghi dữ liệu ADC và ADIF được cài đặt .Trong chế độ chuyển đổi đơn , ADSC được xóa 1 cách đồng thời . Phần mềm sau đó có thể cài đặt ADSC trở lại và 1 quá trình chuyển đổi mới sẽ được khởi tạo trên sùn lên đầu tiên của sùn xung nhịp ADC .

Trong chế độ chạy tự do , 1 quá trình chuyển đổi mới sẽ được bắt đầu ngay lập tức sau khi quá trình chuyển đổi hoàn thành trong khi ADSC vẫn còn lại ở mức cao . Cho 1 sự liệt kê chi tiết về thời gian chuyển đổi xem bảng 95

**Figure 110.** ADC Timing Diagram, First Conversion (Single Conversion Mode)**Figure 111.** ADC Timing Diagram, Single Conversion**Figure 112.** ADC Timing Diagram, Free Running Conversion

**Table 95. ADC Conversion Time**

Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
First conversion	13.5	25
Normal conversions, single ended	1.5	13
Normal conversions, differential	1.5/2.5	13/14

## Các kênh khuyêch đại riêng biệt

Khi sử dụng các kênh khuyêch đại riêng biệt khác nhau , các dạng đã biết của quá trình chuyển đổi cần được tạo ra trong quá trình xem xét .

Các quá trình chuyển đổi riêng biệt thì được đồng bộ hóa đến xung nhịp bên trong CK<sub>ADC2</sub> bằng  $\frac{1}{2}$  chu kì xung nhịp ADC .Quá trình đồng bộ này được thực hiện 1 cách tự động bằng giao diện ADC theo 1 cách mà quá trình lấy mẫu và giữ mức xuất hiện tại 1 sườn xác định của CK<sub>ADC2</sub> . Một quá trình khởi động được khởi tạo bằng người sử dụng (Tất cả các quá trình chuyển đổi đơn , và quá trình chuyển đổi chạy tự do đầu tiên ) . Khi mà CK<sub>ADC2</sub> ở mức thấp sẽ tạo ra các khoảng thời gian giống nhau như là quá trình chuyển đổi single ended (13 xung nhịp ADC từ chu kì xung nhịp được đếm gộp trước tiếp theo ) 1 quá trình chuyển đổi được khởi tạo bởi người sử dụng khi mà CK<sub>ADC2</sub> ở mức cao sẽ mất 14 chu kì xung nhịp từ bộ phận đồng bộ hóa . Trong chế độ chạy tự do 1 quá trình chuyển đổi mới được khởi tạo ngay lập tức sau khi quá trình trước đó được hoàn thành , và do đó CK<sub>ADC2</sub> ở mức cao tại thời điểm này , tất cả các chế độ chạy tự do được khởi động 1 cách tự động sẽ mất 14 chu kì xung nhịp ADC

Cổng khuyêch đại được tối ưu hóa cho dải tần số của 4KHz tại tất cả các chế độ cài đặt khuyêch đại. Các tần số cao hơn có thể được xác định lên quá trình khuyêch đại không tuyến tính . 1 bộ lọc thông thấp (low - pass) lên được sử dụng nếu tín hiệu đầu ra bao gồm các thành phần tần số cao hơn dải sóng ở cổng khuyêch đại chú ý rằng tần số xung nhịp ADC thì phụ thuộc vào giới hạn dải tần của cổng khuyêch đại . Ví dụ : chu kì xung nhịp ADC có thể là 6μs , cho phép 1 kênh được lấy mẫu tại 12kSPS , bất chấp dải tần của kênh này .

## Sự thay đổi kênh hoặc lựa chọn mốc chuẩn

Các bít MUXn và REFS 1:0 trong thanh ghi ADMUX được ghi vào bộ đệm đơn thông qua 1 thanh ghi tạm thời lên cái mà CPU sẽ truy cập ngẫu nhiên . Điều này đảm bảo rằng các kênh và sự lựa chọn mốc chuẩn chỉ có thể xảy ra tại 1 điểm an toàn trong suốt quá trình chuyển đổi . Kênh mà sự lựa chọn mốc chuẩn thì tiếp tục được cập nhật cho đến khi 1 quá trình chuyển đổi được bắt đầu . Mỗi một quá trình chuyển đổi được bắt đầu, các kênh và bộ lựa chọn mốc chuẩn bị khóa để đảm bảo rằng khoảng thời gian lấy mẫu cho ADC là tối ưu . Việc cập nhật liên tiếp khôi phục trong chu kì xung nhịp ADC cuối cùng trước khi quá trình chuyển đổi được hoàn thành (ADIF trong ADCSRA được cài đặt ) . Chú ý rằng quá trình chuyển đổi này bắt đầu trên sườn lên

tiếp theo của xung nhịp ADC sau khi ADSC được viết . Người sử dụng do vậy được khuyên là không được viết kênh mới hoặc các giá trị lựa chọn mốc chuẩn mới lên ADMUX cho đến khi 1 chu kì xung nhịp ADC sau khi ADSC được viết.

Trường hợp đặc biệt này nên được tạo ra khi quá trình chuyển đổi các kênh riêng biệt . Các kênh riêng biệt vừa được lựa chọn , cổng khuyếch đại sẽ tiêu tốn nhiều hơn 125 $\mu$ s để đạt được giá trị ổn định của nó vì vậy quá trình chuyển đổi này không nên được bắt đầu sau khi việc lựa chọn 1 kênh riêng biệt mới . Như 1 sự lựa chọn , kết quả của quá trình này đạt được trong vòng chu kì này lên được bỏ đi .

Thời gian ổn định giống nhau lên được quan sát trong quá trình chuyển đổi riêng biệt đầu tiên sau khi sự thay đổi mốc chuẩn ADC (bằng việc thay đổi các bít REFS 1:0 trong thanh ghi ADMUX )

Nếu như giao diện JTAG được kích hoạt , chức năng của các kênh ADC trên cổng PORT7:4 được ghi đè . Tham khảo bảng 42 trang 83

## Các kênh đầu vào ADC

Khi việc lựa chọn thay đổi các kênh , người sử dụng lên quan sát các hướng dẫn bên dưới để đảm bảo rằng việc lựa chọn các kênh được chính xác trong chế độ chuyển đổi đơn , luôn luôn lựa chọn các kênh trước khi khởi động 1 quá trình chuyển đổi. Sự lựa chọn kênh có thể bị thay đổi trong 1 chu kì xung nhịp ADC sau khi viết là 1 lên bít ADSC . Tuy nhiên , phương pháp đơn giản nhất là đợi cho đến khi quá trình chuyển đổi được hoàn thành trước khi thay đổi lựa chọn kênh . Trong chế độ chạy tự do , luôn lựa chọn các kênh trước khi bắt đầu quá trình chuyển đổi đầu tiên . Sự lựa chọn kênh này phải được thay đổi trong 1 chu kì xung nhịp ADC sau khi viết là 1 lên bít ADSC . Tuy nhiên , Tuy nhiên , phương pháp đơn giản nhất là đợi cho đến khi quá trình chuyển đổi được hoàn thành trước khi thay đổi lựa chọn kênh . Do đó quá trình chuyển đổi tiếp theo sẽ săn sang được bắt đầu 1 cách tự động , kết quả tiếp theo sẽ được phản xạ vào trong lựa chọn kênh trước đó . Quá trình chuyển đổi kế tiếp sẽ phản xạ 1 sự lựa chọn kênh mới

Khi quá trình chuyển mạch tới 1 kênh khuyếch đại riêng biệt, KQ của quá trình chuyển đổi đầu tiên có thể kém chính xác do sự cùn thiết của thời gian lảng trong 1 mạch khóa offset tự động , Người sử dụng lên bỏ qua của quá trình đầu tiên

## Điện áp tham chiếu ADC

Điện áp tham chiếu của ADC ( $V_{REF}$ ) hiển thị 1 dải giá trị của quá trình chuyển đổi cho bộ ADC . Các kênh single ended cái mà vượt trội hơn giá trị

( $V_{REF}$ ) thì sẽ gây ra 1 đoạn mã đóng lên giá trị 0x3FF . ( $V_{REF}$ ) có thể được lựa chọn giữa các giá trị AVcc, điện áp tham chiếu bên trong 2,56V hoặc chân  $V_{REF}$  bên ngoài

AVCC được nối với bộ ADC thông qua 1 bộ chuyển mạch bị động . Điện áp tham chiếu 2,56V bên trong được phát ra từ điện áp tham chiếu bandgap bên trong ( $V_{BG}$ ) thông qua 1 bộ khuyếch đại bên trong , trong trường hợp này chân AREF bên trong được nối trực tiếp đến bộ ADC , và điện áp tham chiếu có thể được miễn nhiệm với nhiều hơn bằng việc nối 1 tụ điện giữa chân AREF và chân GND .  $V_{REF}$  có thể cũng được đo tại chân AREF với 1 vôn kế có trở kháng cao . Chú ý rằng  $V_{REF}$  là 1 nguồn trở kháng cao và chỉ là 1 tải điện dung lên được kết nối vào 1 hệ thống

Nếu người sử dụng có 1 nguồn điện áp ổn định được kết nối đến chân AREF , người sử dụng không thể lựa chọn các điện áp tham chiếu khác , trong các ứng dụng do đó họ sẽ rút ngắn (shorted) điện áp bên ngoài . Nếu không có nguồn điện áp bên ngoài nào được đặt vào chân AREF người sử dụng có thể chuyển mạch giữa AVCC và 2,56V như là 1 lựa chọn điện áp tham chiếu . KQ của quá trình chuyển đổi ADC sau khi việc chuyển mạch nguồn điện áp tham chiếu có thể không chính xác , và người sử dụng được khuyên là không chấp nhận kết quả này

Nếu các kênh khác nhau được sử dụng , điện áp tham chiếu được lựa chọn sẽ bị đóng lên chân AVCC như được hiển thị trong bảng 136 trang 326

## Khóa cắt nhiễu ADC

ADC có 1 khóa cắt nhiễu , cái mà kích hoạt quá trình chuyển đổi trong suốt chế độ ngủ để giảm nhiễu được đưa ra từ lõi của CPU và các ngoại vi IO khác . Khóa cắt nhiễu có thể được sử dụng với chế độ giảm nhiễu ADC và chế độ bận Idle để sử dụng tính năng này , 1 quy trình dưới đây lên được sử dụng

- Đảm bảo rằng bộ ADC được kích hoạt và không có quá trình chuyển đổi nào đang bận . Chế độ chuyển đổi đơn phải được lựa chọn và các ngắt hoàn thành bộ chuyển đổi ADC phải được kích hoạt
- Việc truy nhập vào chế độ giảm nhiễu ADC hoặc chế độ Idle . Bộ ADC sẽ bắt đầu 1 quá trình chuyển đổi mỗi lần mà CPU bị tạm dừng
- Nếu như không có ngắt nào khác xuất hiện trước khi quá trình chuyển đổi ADC hoàn thành các ngắt của ADC sẽ đánh thức CPU và các chương trình con phục vụ ngắt hoàn thành quá trình chuyển đổi ADC . Nếu 1 ngắt khác đánh thức CPU trước khi quá trình

chuyển đổi ADC hoàn thành , ngắt đó sẽ được thực thi , và 1 yêu cầu ngắt hoàn thành chuyển đổi ADC sẽ được sinh ra khi mà quá trình chuyển đổi ADC hoàn thành . CPU sẽ phục hồi trong chế độ hoạt động cho đến khi 1 lệnh sleep mới được thực thi

Chú ý rằng : bộ ADC sẽ không được tắt 1 cách tự động khi truy nhập vào các chế độ ngủ khác 2 chế độ Idle và giảm nhiễu ADC . Người sử dụng được khuyên là nên viết mức Zero lên bít Adle trước khi quá trình truy nhập vào nhiều chế độ ngủ khác nhau để tránh tổn hao điện áp đoán mạch. Nếu như bộ ADC được kích hoạt trong nhiều chế độ ngủ và người sử dụng muốn tiến hành các quá trình chuyển đổi khác , người sử dụng được khuyên là tắt ADC và sau đó đánh thức lại từ chế độ ngủ để yêu cầu 1 quá trình chuyển đổi mở rộng để lấy 1 kết quả hợp lệ

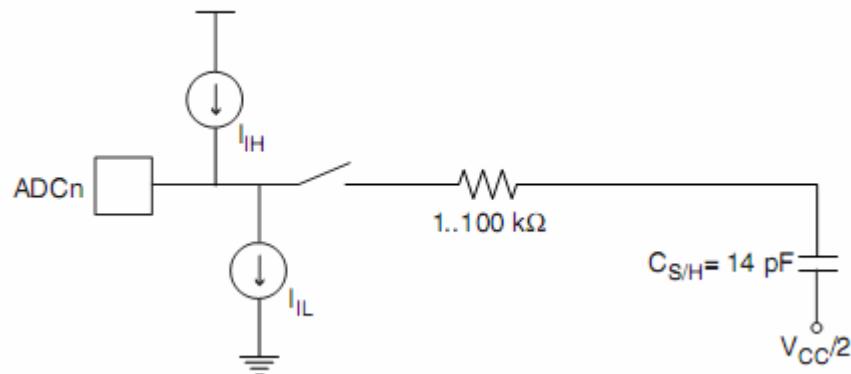
## Mạch đầu vào tương tự

Mạch đầu vào tương tự cho các kênh single ended được minh họa trong hình 113 . 1 nguồn tương tự đặt lên ADCn để subjected lên chân của tụ và input leakage của chân đó , bất chấp việc kênh đó được lựa chọn như là 1 đầu vào cho bộ ADC . Khi 1 kênh được lựa chọn, nguồn phải điều khiển tụ S/H thông qua bộ điện trở (được kết hợp các điện trở ở trong đường dẫn vào ) .

Bộ ADC được tối ưu hóa cho tất cả các tín hiệu tương tự với trở kháng đầu ra khoảng  $10K\Omega$  hoặc nhỏ hơn . Nếu như 1 nguồn được sử dụng , thời gian lấy mẫu sẽ không đáng kể . Nếu 1 nguồn với trở kháng cao hơn được sử dụng , thời gian lấy mẫu sẽ phụ thuộc vào khoảng thời gian mà nguồn đó cần để nạp vào tụ S/H , và có thể biến đổi rộng . Người sử dụng được khuyến cáo chỉ sử dụng các nguồn trở kháng thấp với các tín hiệu biến đổi chậm , do đó điều này làm cực tiêu thời gian nạp vào tụ S/H .

Nếu các kênh có độ khuyếch đại khác được sử dụng , mạch đầu ra sẽ nhìn thấy 1 số điểm khác biệt mặc dù các nguồn trở kháng ở 1 vài trăm  $K\Omega$  hoặc nhỏ hơn được khuyên dùng

Các thành phần của tín hiệu cao hơn tần số Nyquist ( $f_{ADC}/2$ ) nên không được đưa ra cho các loại kênh khác . Để tránh biến dạng méo từ unpredictable signal convolution . Người sử dụng được khuyên lén gỡ bỏ các thành phần tần số cáo bằng 1 bộ lọc thông thấp trước khi việc áp dụng các tín hiệu như là các đầu vào tới bộ ADC

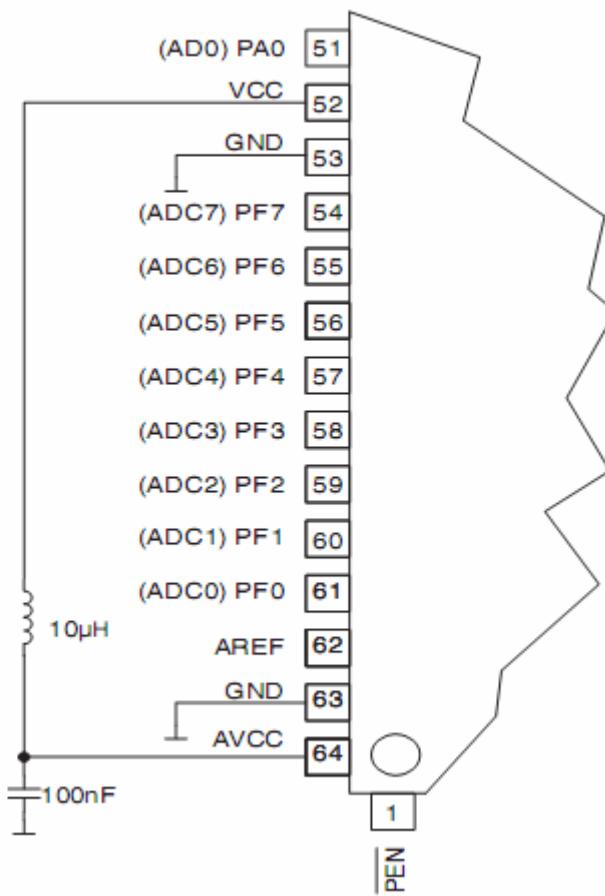
**Figure 113. Analog Input Circuitry**

### Công nghệ khóa cắt nhiễu Analog

Mạch số bên trong và bên ngoài của thiết bị phát ra EMI cái mà có thể ảnh hưởng tới độ chính xác của bộ do tương tự . Nếu độ chính xác của quá trình chuyển đổi là quan trọng , mức nhiễu có thể được giảm bằng cách áp dụng kỹ thuật dưới đây :

- Giữ cho các đường dẫn tín hiệu tương tự ngắn nhất có thể . Đảm bảo rằng các Analog track chạy qua mặt đất , và giữ chúng tránh xa bộ chuyển mạch digital track tốc độ cao
- Chân AVCC trên thiếp bị nên được kết nối với điện áp nguồn cấp số Vcc như là 1 mạng NC được chỉ ra trong hình 114
- sử dụng chức năng khóa cắt nhiễu để làm giảm nhiễu từ CPU
- Nếu bất cứ chân cổng nào của ADC được sử dụng như là 1 đầu ra số điều thiết yếu là sẽ không có sự chuyển mạch nào trong khi 1 quá trình chuyển đổi đang tiến hành

Figure 114. ADC Power Connections



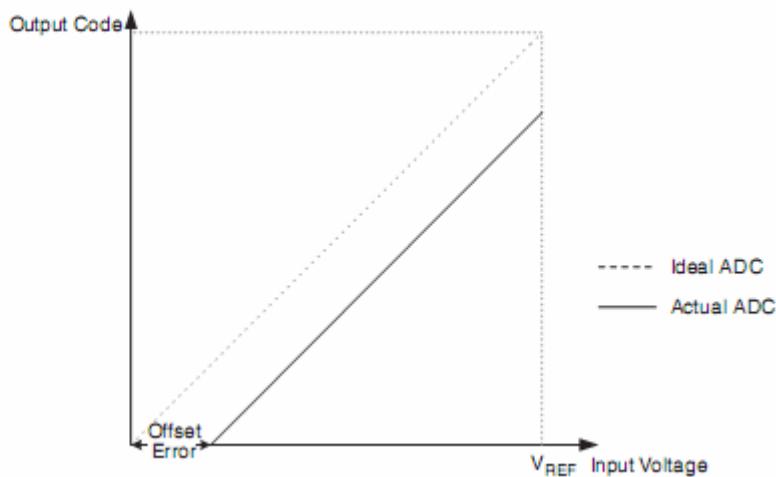
## Mạch bù Offset

Cổng khuyếch đại có 1 mạch khử build-in offset , cái mà triệt tiêu offset của các bộ đo riêng biệt nhiều nhất có thể . Các offset còn lại trong đường dẫn tương tự có thể được đo trực tiếp bằng việc lựa chọn các kênh giống nhau cho cả 2 loại đầu vào riêng biệt . Các phần dư offset này có thể được loại trừ bằng phần mềm khỏi kết quả của việc đo . Việc sử dụng loại phần mềm sửa lỗi offset cơ bản này , offset và các kênh khác có thể được làm giảm bên dưới 1 LSB

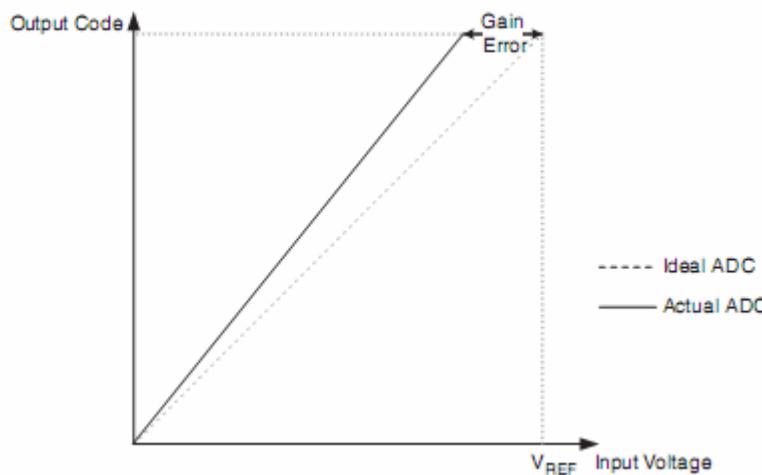
## Xác định độ chính xác của ADC

1 n-bit single ended ADC chuyển đổi 1 dải điện áp giữa chân GND và chân  $V_{REF}$  trong  $2^n$  bậc (LSBs) . Đoạn mã thấp nhất được đọc là 0 , đoạn mã cao nhất đọc là  $2^{n-1}$  .

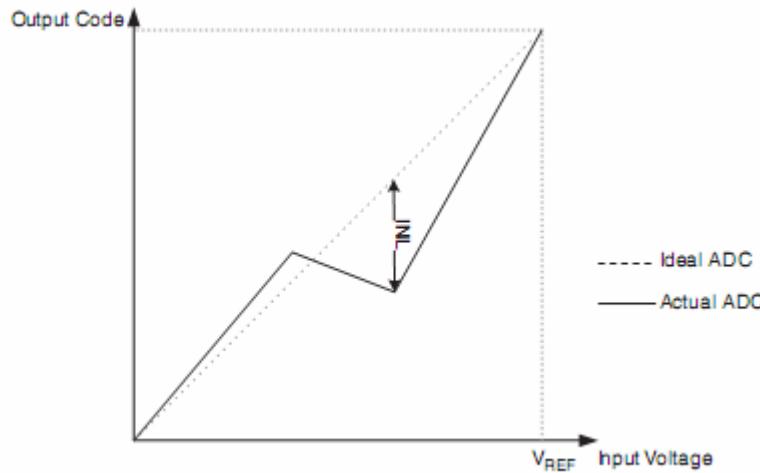
- Hàng loạt tham số được miêu tả độ lệch lý tưởng từ quá trình tiến hành
- Offset : độ lệch của quá trình di chuyển đầu tiên ( từ 0x000 đến 0x001 ) được so sánh với quá trình di chuyển lí tưởng (ở 0,5 LSB) . Giá trị lí tưởng là 0 LSB

**Figure 115. Offset Error**

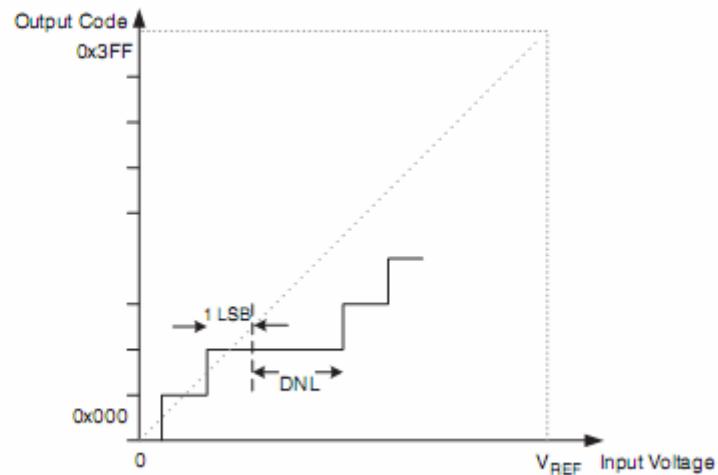
- Lỗi khuyếch đại : Sau khi quá trình điều khiển Offset lỗi khuyếch đại được tìm thấy trong độ lệch di chuyển cuối cùng (0x3FE đến 0x3FF) được so sánh với sự di chuyển lí tưởng (ở 1,5 LSB bên dưới mức cực đại). Giá trị lí tưởng là 0 LSB

**Figure 116. Gain Error**

- Integral non-linearity (INL) : sau quá trình điều chỉnh của Offset, và lỗi khuyếch đại , INL là độ lệch cực đại của quá trình di chuyển hiện thời được so sánh với 1 quá trình di chuyển lí tưởng . Giá trị lí tưởng là 0LSB

**Figure 117. Integral Non-linearity (INL)**

- Differential non-linearity (DNL) : độ lệch cực đại của actual code width (đoạn giữa 2 sự di chuyển kế cận ) từ ideal code width . Giá trị lí tưởng là 0LSB

**Figure 118. Differential Non-linearity (DNL)**

Lỗi lượng tử hóa : dù cho sự lượng tử hóa của điện áp đầu vào là 1 số hữu hạn của các mã , 1 khoảng giá trị của điện áp đầu vào (1LSB) sẽ viết các giá trị giống nhau : luôn luôn là  $\pm 0,5$  LSB .

Độ chính xác tuyệt đối : Độ lệch cực đại của 1 quá trình di chuyển thực sự (không có điều chỉnh ) được so sánh với 1 sự di chuyển lí tưởng cho bất cứ mã nào . Điều này là ảnh hưởng phức tạp của Offset lỗi khuyết ch đại , các lỗi khác Non-linearity , lỗi lượng tử hóa . Giá trị lí tưởng là  $\pm 0,5$  LSB

## Kết quả của quá trình chuyển đổi ADC

Sauk hi quá trình chuyển đổi được hoàn tất (ADIF ở mức cao) , KQ của quá trình chuyển đổi có thể được tìm thấy ở trong thanh ghi Kết quả ADC (ADCH và ADCL )

Với quá trình chuyển đổi single ended , kết quả là :

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

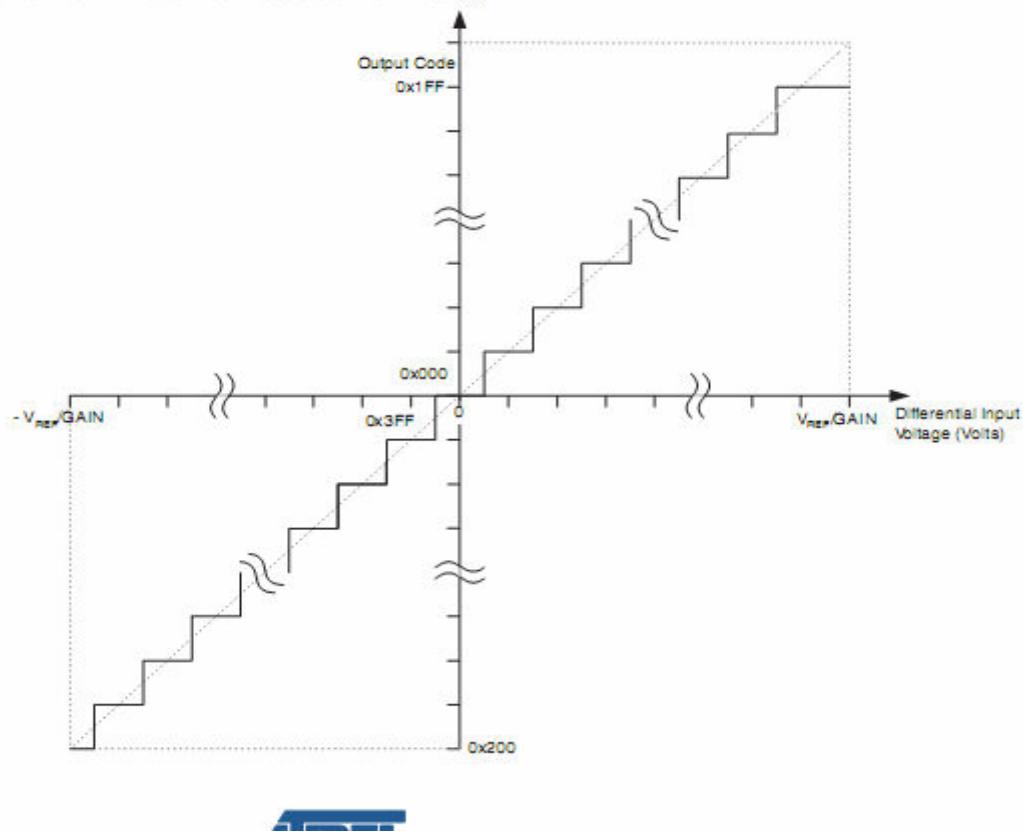
Nơi mà  $V_{IN}$  là điện áp trên chân đầu vào được lựa chọn và  $V_{REF}$  được lựa chọn từ điện áp tham chiếu (Xem bảng 97 . 98 trang 242) . Bít 0x000 được nối với đất và 0x3FF được nối với điện áp tham chiếu được lựa chọn nhỏ hơn 1 LSB .

Nếu các kênh khác được sử dụng kết quả là :

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot GAIN \cdot 512}{V_{REF}}$$

Nơi mà  $V_{POS}$  là điện áp trên chân đầu vào dương ,  $V_{NEG}$  là điện áp trên chân đầu vào âm GAIN được lựa chọn là tỉ lệ khuyế ch đại , và  $V_{REF}$  được lựa chọn điện áp tham chiếu . Kết quả được đưa ra bằng 2 dạng , từ 0x200 (-512d) thông qua 0x1FF (+ 511d) . Chú ý rằng nếu người sử dụng muốn tiến hành 1 quá trình kiểm tra nhanh của kết quả , nó phải được đủ để đọc MSB của kết quả bít cao nhất (ADC9 trong ADCH) . Nếu bít này được viết là 1 kết quả là âm , nếu nó là 0 kết quả là dương . Hình 119 chỉ ra bộ giải mã của dải tín hiệu đầu vào khác nhau

Bảng 96 chỉ ra kết quả của các mã đầu vào , nếu như các cặp kênh đầu vào khác nhau (ADCn – ADCm) được lựa chọn với 1 hệ số khuyế ch đại GAIN lớn và điện áp tham chiếu  $V_{REF}$

**Figure 119.** Differential Measurement Range**Table 96.** Correlation Between Input Voltage and Output Codes

$V_{ADCn}$	Read code	Corresponding decimal value
$V_{ADCm} + V_{REF}/GAIN$	0x1FF	511
$V_{ADCm} + 511/512 V_{REF}/GAIN$	0x1FF	511
$V_{ADCm} + 511/512 V_{REF}/GAIN$	0x1FE	510
...	...	...
$V_{ADCm} + 1/512 V_{REF}/GAIN$	0x001	1
$V_{ADCm}$	0x000	0
$V_{ADCm} - 1/512 V_{REF}/GAIN$	0x3FF	-1
...	...	...
$V_{ADCm} - 511/512 V_{REF}/GAIN$	0x201	-511
$V_{ADCm} - V_{REF}/GAIN$	0x200	-512

Example:

$ADMUX = 0xED$  (ADC3 - ADC2, 10x gain, 2.56V reference, left adjusted result)

Voltage on ADC3 is 300 mV, voltage on ADC2 is 500 mV.

$ADCR = 512 * 10 * (300 - 500) / 2560 = -400 = 0x270$

ADCL will thus read 0x00, and ADCH will read 0x9C. Writing zero to ADLAR right adjusts the result:  $ADCL = 0x70$ ,  $ADCH = 0x02$ .

## Chương trình mẫu

### Thanh ghi lựa chọn ghép kênh ADC\_ADMUX

Bit	7	6	5	4	3	2	1	0	ADMUX
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bits7:6\_REFs 1: 0 . Các bít lựa chọn điện áp tham chiếu

Các bít này lựa chọn điện áp tham chiếu cho bộ ADC , như được chỉ ra trong bảng 97 . Nếu các bít này bị thay đổi trong suốt quá trình chuyển đổi , sự thay đổi đó sẽ không có ảnh hưởng cho đến khi quá trình chuyển đổi được hoàn thành (ADIF trong thanh ghi ADCSRA được cài đặt) . Các điện áp tham chiếu bên trong có thể sử dụng nếu 1 điện áp ham chiếu bên ngoài đang được đặt lên chân AREF

**Table 97. Voltage Reference Selections for ADC**

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

Bít 5\_ANAR : kết quả điều chỉnh di chuyển ADC

Bít ADNAR ảnh hưởng với sự biểu diễn của kết quả sự chuyển đổi ADC trong thanh ghi dữ liệu ADC . Viết mực 1 lên ADNAR để điều chỉnh di chuyển sang trái kết quả . Nói cách khác kết quả này được điều chỉnh đúng . Việc thay đổi bít ADNAR sẽ ảnh hưởng đến thanh ghi dữ liệu ADC ngay lập tức , bất chấp bất cứ quá trình chuyển đổi nào đang hoạt động . Để có 1 sự miêu tả hoàn chỉnh về bít này xem thanh ghi dữ liệu ADC\_ADCH và ADCL ở trang 245

Bít 4..0\_ MUX4 : 0 – bít lựa chọn hệ số khuyếch đại và tương tự .

Giá trị của các bít này lựa chọn sự kết hợp của các đầu vào tương tự được kết nối tới ADC . Các bít này cũng lựa chọn các hệ số khuyếch đại cho các kênh khác nhau . Xem bảng 98 . Nếu các bít này bị thay đổi trong suốt quá trình chuyển đổi . Sự thay đổi sẽ không có hiệu lực cho đến khi quá trình chuyển đổi này được hoàn thành (ADIF trong ADCSRA được cài đặt )

Table 98. Input Channel and Gain Selections

MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0			
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000 <sup>(1)</sup>		ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010 <sup>(1)</sup>		ADC0	ADC0	200x
01011		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110		ADC2	ADC2	200x
01111		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x

### Thanh ghi trạng thái và điều khiển ADC\_ADCSRA

Bit	7	6	5	4	3	2	1	0	ADCSRA
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

Bits7\_ADEN : Kích hoạt ADC

Việc đặt bit này =1 sẽ kích hoạt ADC . Nếu =0 thì ADC bị tắt . Quá trình điều khiển ADC tắt trong khi 1 sự chuyển đổi đang tiến hành sẽ kết thúc quá trình chuyển đổi này .

Bít 6 \_ADSC : Bắt đầu quá trình chuyển đổi ADC

Trong chế độ chuyển đổi đơn , viết bit này là 1 để khởi động mỗi quá trình chuyển đổi . Trong chế độ chạy tự do viết bit này là 1 để khởi động quá trình chuyển đổi đầu tiên . Quá trình chuyển đổi đầu tiên sau khi bit ADSC được viết sau khi ADC

được kích hoạt , hoặc ADSC được viết cùng thời điểm với ADC được kích hoạt sẽ lấy mất 25 chu kỳ xung nhịp ADC thay thế cho chế độ thông thường là 13 . Quá trình chuyển đổi đầu tiên này tiến hành khởi tạo ADC

ADSC sẽ đọc là 1 chỉ cần 1 quá trình đang tiến hành . Khi quá trình chuyển đổi được hoàn thành nó sẽ trở về không . Việc viết là 0 lên bít này thì sẽ không có hiệu lực gì

#### Bít 5\_ADSR : Lựa chọn chế độ chạy tự do ADC

Khi bít này được viết là 1 ADC hoạt động trong chế độ chạy tự do . Trong chế độ này , ADC lấy mẫu và cập nhật thanh ghi dữ liệu liên tiếp . Viết bít này là 0 sẽ kết thúc chế độ chạy tự do .

#### Bít 4\_ADIF : Cờ báo ngắt ADC

Bít này được cài đặt khi 1 quá trình chuyển đổi ADC được hoàn thành và các thành ghi dữ liệu được cập nhật các ngắt báo hoàn thành quá trình chuyển đổi ADC được thực thi nếu bít ADIE và bít I trong thanh ghi SREG được đặt . ADIF bị xóa bằng phần cứng khi thực thi xong các vector điều khiển ngắt tương ứng . Như 1 sự lựa chọn , ADIF bị xóa bằng việc viết mức logic 1 lên cờ . Chú ý rằng nếu đang thực hiện 1 quá trình Read-modify-Write trên thanh ghi ADCSRA , 1 ngắt đang tiến hành có thể bị vô hiệu hóa . Điều này cũng xảy ra nếu các lệnh SBI và CBI được sử dụng

#### Bít 3\_ADIE : Kích hoạt các ngắt ADC

Khi bít này là 1 và bít I trong thanh ghi SREG được đặt thì cờ báo ngắt hoàn thành quá trình chuyển đổi ADC được kích hoạt

#### Bít 2..0\_APDS 2:0 . Bít lựa chọn bộ đếm gộp trước ADC

Các bít này xác định hệ số chia giữa tần số XTAL và các xung nhịp đầu vào lên bộ ADC

Table 99. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

## Thanh ghi dữ liệu ADC – ADCH , ADCL

**ADLAR = 0:**

Bit	15	14	13	12	11	10	9	8	ADCH
	-	-	-	-	-	-	ADC9	ADC8	ADCL
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	
	7	6	5	4	3	2	1	0	
ReadWrite	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

**ADLAR = 1:**

Bit	15	14	13	12	11	10	9	8	ADCH
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCL
	ADC1	ADC0	-	-	-	-	-	-	
	7	6	5	4	3	2	1	0	
ReadWrite	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Khi quá trình chuyển đổi ADC được hoàn thành , kết quả được tìm thấy ở trong 2 thanh ghi . Nếu 2 kênh riêng biệt được sử dụng kết quả được đưa ra trong 2 dạng hoàn toàn khác nhau .

Khi ADCL được đọc thanh ghi dữ liệu ADC không được cập nhật cho đến khi ADCH được cập nhật . Nếu kết quả được di chuyển và không cần độ chính xác 8 bit , nó đủ điều kiện để đọc ADCH . Nói cách khác ADCL phải được đọc đầu tiên sau đó là ADCH .

Bít ADLAR trong thanh ghi ADMUX và các bít MUXn trong thanh ghi ADMUX ảnh hưởng lên cách mà kết quả được đọc từ các thanh ghi . Nếu ADLAR được đặt kết quả được điều chỉnh di dời . Nếu ADLAR bị xóa (như mặc định ) . Kết quả là đúng

Bít ADC9:0 . Kết quả của quá trình chuyển đổi .

Các bít này đưa ra kết quả của quá trình chuyển đổi như được đưa ra ở bảng 241

## XXII . Giao diện JTAG và hệ thống hiệu chỉnh lỗi trên chip

### Đặc điểm

- giao diện JTAG ( phù hợp với tiêu chuẩn IEEE 1149 )
- khả năng Quét giới hạn biên (boundary – scan ) theo chuẩn IEEE1149 (JTAG)
- Bộ hiệu chỉnh lỗi truy nhập tới
  - + Tất cả các thành phần ngoại vi bên trong
  - + RAM trong và ngoài
  - + File thanh ghi bên trong
  - + bộ đếm chương trình
  - + Các bộ nhớ EEPROM và bộ nhớ chương trình
- Hỗ trợ hiệu chỉnh lỗi trên chip mở rộng cho các điều kiện ngắt (break) bao gồm
  - + lệnh ngắt AVR
  - + dừng trên quá trình thay đổi của dòng nhớ chương trình
  - + Bước dừng đơn (Single Step Break )
  - + lập trình điểm ngắt bộ nhớ trên địa chỉ đơn hoặc dài địa chỉ
- Lập trình Flash , EEPROM , cầu chì , và các bit khóa thông qua giao diện JTAG
- Đã hỗ trợ việc hiệu chỉnh trên Chip bởi AVR studio

### Tổng quan

AVR phù hợp với tiêu chuẩn IEEE 1149 giao diện JTAG có thể được sử dụng cho :

- quá trình kiểm tra PCBs bằng cách sử dụng khả năng quét biên JTAG
- lập trình các bộ nhớ không thể thay đổi , các bit cầu chì và bit khóa
- Quá trình hiệu chỉnh trên chip

Một sự miêu tả ngắn gọn được đưa ra trong các phần bên dưới . Các sự miêu tả chi tiết sau cho việc lập trình giao diện JTAG , và sử dụng chuỗi quét biên có thể được tìm thấy trong phần “Programming Via the JTAG Interface “ trên trang 305 và “IEEE 1149.1(JTAG) Boundary scan “trên trang 252 . Sự hỗ trợ hiệu chỉnh lỗi trên chip đang được xét đến các lệnh JTAG riêng , và được phân phối trong khuôn khổ ATMEL và chỉ được lựa chọn 3 đại lý cung cấp

Hình 120 chỉ ra một sơ đồ khái của giao diện JTAG và hệ thống hiệu chỉnh lỗi trên chip. Bộ điều khiển TAP là một bộ máy trạng thái (state machine ) được điều khiển bởi các tín hiệu TCK và TMS . Bộ điều khiển TAP lựa chọn hoặc là thanh ghi lệnh JTAG hoặc là 1 trong số rất nhiều thanh ghi dữ liệu như là một

chuỗi quét (Shift Register) giữa đầu vào TDI và đầu ra TDO . Thanh ghi lệnh giữ các lệnh JTAG điều khiển sự tiến hành xử lí của một thanh ghi dữ liệu .

Thanh ghi ID , thanh ghi Bypass , và chuỗi quét biên là các thanh ghi dữ liệu được sử dụng cho quá trình kiểm tra board – level . Lập trình giáo diện JTAG (thực sự bao gồm nhiều thanh ghi vật lý và thanh ghi dữ liệu ảo ) được sử dụng cho lập trình nối tiếp thông qua giao diện JTAG . Các chuỗi quét và điểm dừng chuỗi quét bên trong chỉ được sử dụng cho việc hiệu chỉnh lỗi trên chip.

## Cổng truy nhập kiểm tra – Test Access Port – TAP

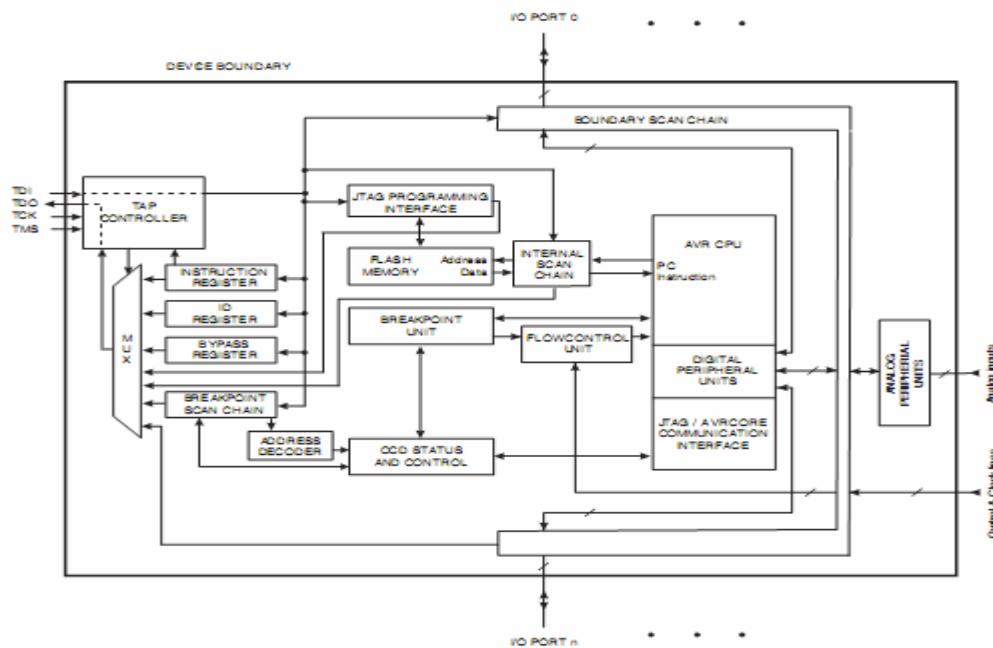
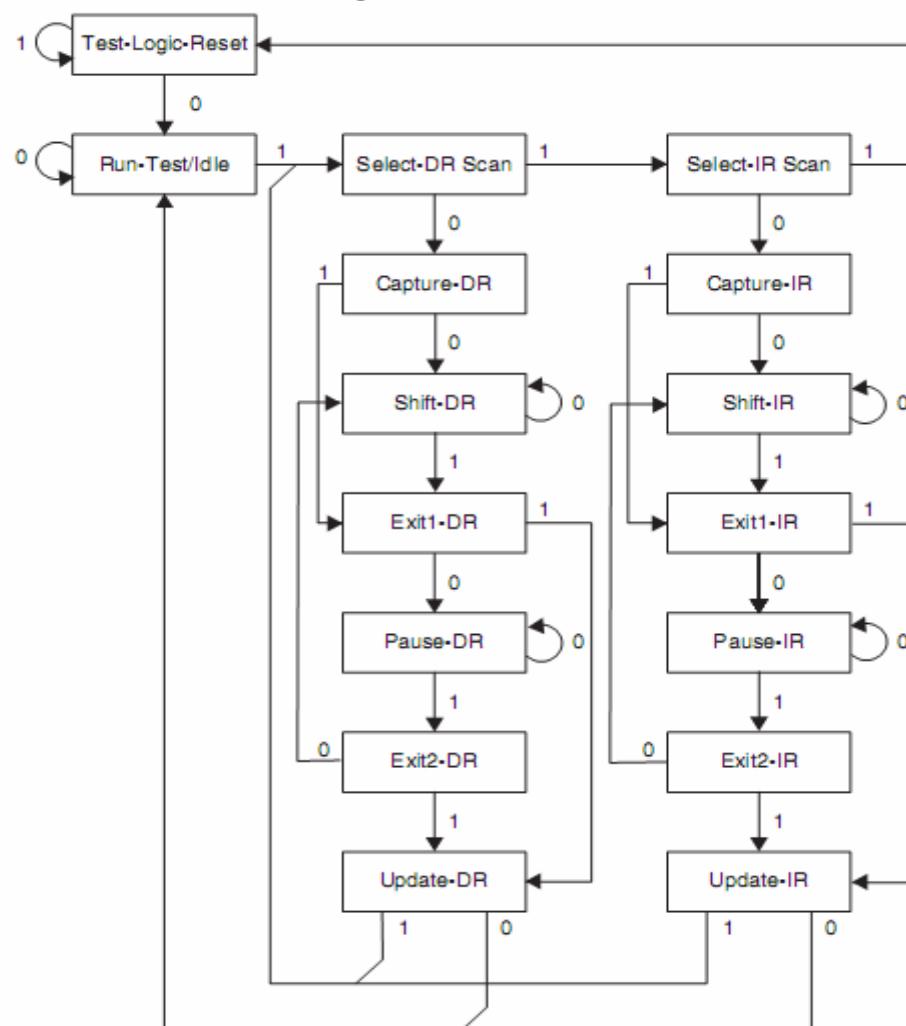
Giao diện JTAG được truy nhập thông qua 4 chân của AVR . Trong công nghệ của AVR , các chân này cấu thành cổng truy nhập kiểm tra – TAP . các chân đó là :

- TMS : lựa chọn chế độ kiểm tra . Chân này được sử dụng cho việc định hướng thông qua thiết bị trạng thái bộ điều khiển
- TCK : kiểm tra xung nhịp . Hoạt động của JTAG thì đồng bộ hóa lên chân TCK
- TDI : kiểm tra dữ liệu vào . dữ liệu cổng vào nối tiếp được di chuyển vào trong thanh ghi lệnh hoặc thanh ghi dữ liệu (chuỗi quét )
- TDO : kiểm tra dữ liệu ra . Dữ liệu đầu ra nối tiếp từ thanh ghi lệnh và thanh ghi dữ liệu

Tiêu chuẩn IEEE 1149 cũng xác định một lựa chọn tín hiệu TAP , TRST – Test ReSeT – cái mà không được cung cấp .

Khi mà cầu chì JTAGEN không được lập trình , 4 chân TAP này là chân cổng bình thường và bộ điều khiển TAP thì trong chế độ reset . Khi được lập trình và bit JTD trong thanh ghi MCUCSR bị xóa ,các tín hiệu đầu vào TAP được kéo vào trong ở mức cao và JTAG được kích hoạt cho chuỗi quét biên (boundary – scan ) và quá trình lập trình . Trong trường hợp này , đầu ra TAP là (TDO ) được di chuyển thay đổi trong trạng thái mà ở đó bộ điều khiển JTAG TAP đang không di chuyển dữ liệu , và vì vậy phải được nối đến một điện trở pull-up hoặc phần cứng khác có tính năng pull-up (chẳng hạn như đầu vào TDI của thiết bị tiếp theo trong chuỗi quét ) . Thiết bị được di chuyển với cầu chì được lập trình .

Về hệ thống sửa lỗi trên chip , thêm vào tới các chân giao diện JTAG , chân RESET được quan sát bởi bộ hiệu chỉnh lỗi có thể dò tới các nguồn Reset bên ngoài . Bộ hiệu chỉnh lỗi cũng có thể kéo chân RESET xuống mức thấp để reset toàn bộ hệ thống , giả sử rằng chỉ mở các collector trên đường Reset là được sử dụng trong ứng dụng này .

**Figure 120.** Block Diagram**Hình 121 :** sơ đồ trạng thái bộ điều khiển TAP**Figure 121.** TAP Controller State Diagram

## Bộ điều khiển TAP

Bộ điều khiển TAP là một cơ cấu trạng thái hữu hạn 16 trạng thái mà điều khiển hoạt động của mạch chuỗi quét biên , mạch lập trình JTAG , hoặc hệ thống sửa lỗi trên chip . Sự chuyển đổi trạng thái được mô tả trong hình 121 phụ thuộc vào tín hiệu được đưa ra trên chân TMS ( chỉ ra bên cạnh mỗi sự thay đổi trạng thái ) tại thời điểm của sùn lện TCK . Trạng thái khởi tạo sau đó một Reset Power-on là Reset kiểm tra logic

Như được định nghĩa trong tài liệu này , LSB được di chuyển vào trong và ra ngoài trước tiên cho tất cả các thanh ghi Shift

Giá thiết rằng Run-Test/Idle là trạng thái được đưa ra , Một chuỗi sự kiện bình thường cho việc sử dụng giao diện JTAG là :

- Tại đầu vào TMS , áp dụng liên tiếp 1, 1, 0 , 0 tại các sùn lện của TCK để nhập vào thanh ghi lệnh Shift – Shift - I state . Trong khi trong trạng thái này , di chuyển 4 bit của lệnh JTAG vào trong thanh ghi lệnh JTAG từ đầu vào TDI tại sùn lện cẩu TCK . Đầu vào TMS phải được giữ ở mức thấp trong suốt đầu vào cẩu 3 LSB theo thứ tự còn lại trong trạng thái Shift – IR . MSB của lệnh được di chuyển trong khi trạng thái này được di chuyển bằng cách cài đặt TMS ở mức cao . Trong khi lệnh được di chuyển từ trong TDI , trạng thái IR-state đã bắt được di chuyển ra ngoài trên chân TDO . Lệnh JTAG lựa chọn thanh ghi dữ liệu riêng như đường dẫn giữa TDI và TDO và điều khiển các mạch xung quanh thanh ghi được lựa chọn
- Áp dụng liên tiếp 1, 1, 0 để truy nhập lại vào trạng thái Run-Test/Idle . Lệnh được chốt trong đầu ra song song từ đường dẫn thanh ghi Shift trong trạng thái Update-IR . Trạng thái Exit-IR , Pause –IR , và Exit2-IR chỉ được sử dụng cho việc định hướng cơ cấu lựa chọn trạng thái
- Tại đầu vào TMS , đặt liên tiếp 1 , 0 , 0 tại sùn lện của xung TCK để truy nhập thanh ghi dữ liệu Shift – Shift-DR . Trong trạng thái này , cập nhật thanh ghi dữ liệu được lựa chọn (được lựa chọn bằng lệnh JTAG )được đưa ra trong thanh ghi lệnh JTAG )từ đầu vào TDI tại sùn lện của TCK . Để mà giữ nguyên trong trạng thái Shift-DR , đầu vào TMS phải được giữ thấp trong suốt đầu vào của tất cả các bít ngoại trừ MSB . MSB của dữ liệu được di chuyển trong khi trạng thái này được di chuyển bằng việc cài đặt TMS ở mức cao . Trong khi thanh ghi dữ liệu được di chuyển khỏi chân TDI , các đầu vào song song đến thanh ghi dữ liệu đã truy bắt trong trạng thái Capture-DR được di chuyển ra ngoài trên chân TDO
- Áp dụng TMS liên tiếp 1, 1 ,0 để truy nhập lại trạng thái Run-Test/Idle . Nếu thanh ghi dữ liệu đã lựa chọn có một đầu ra song song được chốt , quá trình chốt sẽ xảy ra trong trạng thái Update-DR . Các trạng thái Exit-IR , Pause –IR , và Exit2-IR chỉ được sử dụng cho việc định hướng cơ cấu trạng thái .

Như được chỉ ra trong sơ đồ trạng thái , trạng thái Run-Test/Idle không cần được truy nhập giữa việc lựa chọn lệnh JTAG và việc sử dụng các thanh ghi dữ liệu , và vài lệnh JTAG có thể lựa chọn các hàm chắc chắn để tiến hành xử lý trong trạng thái Run-Test/Idle , làm nó không thích hợp như là một trạng thái Idle

Chú ý : phụ thuộc vào việc khởi tạo của bộ điều khiển TAP , trạng thái Test-Logic-Reset có thể luôn được truy nhập bằng việc giữ TMS ở mức cao trong 5 chu kỳ xung nhịp TCK

Để thêm thông tin chi tiết về các đặc điểm của JTAG , tham khảo thêm phần "bibliography" trang 251

### Sử dụng chuỗi quét biên (Boundary –scan chain )

Một sự miêu tả hoàn thiện của khả năng quét biên được đưa ra trong phần “IEEE 1149.1 (JTAG) Boundary –scan . trên trang 252

### Sử dụng hệ thống hiệu chỉnh lỗi trên chip

Như đã chỉ ra trong hình 120 phần cứng hỗ trợ cho quá trình hiệu chỉnh lỗi trên chip bao gồm các thành phần chính sau :

- 1 chuỗi quét (scan chain) trên giao diện giữa CPU AVR bên trong và các thành phần ngoại vi bên ngoài .
- Thành phần điểm dừng (break point )
- Giao diện truyền thông giữa CPU và hệ thống JTAG .

Tất cả các quá trình đọc hoặc sửa đổi /ghi cần thiết cho quá trình thực thi của bộ hiệu chỉnh lỗi đều được thực hiện bằng việc đặt các lệnh AVR thông qua chuỗi quét AVR CPU . CPU gửi kết quả đến một bộ nhớ I/O được định vị trong vùng nhớ mà là 1 phần của giao diện truyền thông giữa CPU và hệ thống JTAG

Thành phần điểm dừng (Break point ) xử lý dừng trên điểm thay đổi của dòng chương trình , bước dừng đơn , 2 điểm dừng bộ nhớ chương trình , và 2 điểm dừng được nối . Cùng với đó , 4 điểm dừng cũng có thể được cấu hình như sau :

- 4 điểm dừng bộ nhớ chương trình đơn
- 3 điểm dừng bộ nhớ chương trình đơn + 1 điểm dừng bộ nhớ dữ liệu đơn
- 2 điểm dừng bộ nhớ chương trình đơn +2 điểm dừng bộ nhớ dữ liệu đơn
- 2 điểm dừng bộ nhớ dữ liệu đơn +1 điểm dừng bộ nhớ dữ liệu với mask (khoảng điểm dừng )
- 2 điểm dừng bộ nhớ dữ liệu đơn + 1 điểm dừng bộ nhớ dữ liệu với mask (khoảng điểm dừng )

Một bộ hiệu chỉnh lỗi , giống như AVR studio , tuy có thể sử dụng một hoặc nhiều hơn các nguồn tài nguyên cho các mục đích bên trong nó , truyền tải dễ dàng đến người sử dụng cuối

Một danh sách của các đặc tính hiệu chỉnh trên chip của các lệnh JTAG được đưa ra trong “On-chip Debug Specific JTAG Instructions “ trang 250

Cầu chì JTAGEN phải được lập trình để kích hoạt cổng truy nhập kiểm tra JTAG . Thêm vào đó , cầu chì OCDEN phải được lập trình và không có bit khóa nào phải được cài đặt cho hệ thống hiệu chỉnh lỗi trên chíp để hoạt động . Như một tính năng bảo mật , hệ thống sửa lỗi trên chíp bị vô hiệu hóa khi bất cứ bit khóa nào được cài đặt . Nói cách khác , hệ thống sửa lỗi trên chíp sẽ cung cấp 1 cửa sau (back-door) vào trong thiết bị cố định (secured device )

AVR studio kích hoạt một người sử dụng để điều khiển đầy đủ quá trình thực thi của chương trình trên một thiết bị AVR với tính năng sửa lỗi trên chíp , Bộ mô phỏng mạch điện trong AVR , hoặc bộ mô phỏng cài đặt lệnh buil-in trong AVR . AVR Studio hỗ trợ nguồn thực thi của chương trình Assembly được thực thi với Atmel Corporation’s AVR Assembler và các chương trình C trình biên dịch với trình các trình biên dịch Vendor

AVR studio chạy dưới các hệ điều hành của window như XP , NT , 2000

Về một sự miêu tả đầy đủ cho AVR Studio , làm ơn tham khảo thêm phần hướng dẫn sử dụng AVR Studio có thể tìm thấy trên mục online help trong phần mềm AVR Studio . Chỉ có các điểm quan trọng được đưa ra trong tài liệu này .

Tất cả các lệnh thực thi cần thiết thì đều có thể sử dụng trong AVR Studio , cả hai nguồn level và không phải Assembly . Người sử dụng có thể thực thi chương trình , bước đơn thông qua đoạn mã bằng việc vẽ vào trong (tracing) hoặc nhảy qua các chức năng ,

### **Các lệnh hiệu chỉnh xác định JTAG trên chíp**

Hiệu chỉnh trên chíp hỗ trợ được coi như các lệnh JTAG chuyên dụng và được sử dụng trong khuôn khổ của ATTEL và để lựa chọn third-party vendors . Các lệnh được liệt kê cho sự tham khảo

PRIVATE0;\$8 chip	lệnh JTAG riêng cho sự truy nhập hệ thống hiệu chỉnh lỗi trên chíp
PRIVATE1;\$9 chip	lệnh JTAG riêng cho sự truy nhập hệ thống hiệu chỉnh lỗi trên chíp
PRIVATE2;\$A chip	lệnh JTAG riêng cho sự truy nhập hệ thống hiệu chỉnh lỗi trên chíp
PRIVATE3;\$B chip	lệnh JTAG riêng cho sự truy nhập hệ thống hiệu chỉnh lỗi trên chíp

## Hiệu chỉnh lỗi trên chip liên quan đến thanh ghi trong vùng nhớ I/O

### Thanh ghi hiệu chỉnh trên chip – OCDR

Bit	7	6	5	4	3	2	1	0	
Read/Write	R/W	OCDR							
Initial Value	0	0	0	0	0	0	0	0	

Thanh ghi OCDR

cung cấp một kênh truyền thông từ chương trình đang chạy trong vi điều khiển đến bộ hiệu chỉnh lỗi .CPU có thể chuyển một byte dữ liệu đến bộ hiệu chỉnh lỗi bằng việc viết lên vùng nhớ này .Tại cùng thời điểm này ,một cờ bên trong ,thanh ghi hiệu chỉnh I/O diri – IDRDIR được cài đặt đến bộ hiệu chỉnh mà thanh ghi được viết . Khi CPU đặt thanh ghi OCDR 7LSP sẽ từ thanh ghi OCDR .Trong khi MSB là bít IDRDIR. Bộ hiệu chỉnh xóa bít IDRDIR khi nó vừa đọc thông tin

Trong một vài thiết bị AVR ,thanh ghi này được chia sẻ với một vùng nhớ I/O chuẩn .Trong trường hợp này thanh ghi OCDR có thể chỉ được truy cập nếu cầu chì ODCEN được lập trình ,và bộ hiệu chỉnh kích hoạt sự truy nhập đến thanh ghi OCDR .Trong tất cả các trường hợp khác ,vùng nhớ I/O chuẩn được truy nhập .

Tham khảo tài liệu hướng dẫn sử dụng bộ hiệu chỉnh, để biết thêm thông tin về các sử dụng thanh ghi này .

### Sử dụng tính năng lập trình JTAG .

Việc lập trình của các phần AVR thông qua giao diện JTAG được tiến hành thông qua 4 chân cổng JTAG, TCK ,TMS, TDI,TDO. Chỉ có các chân cổng mà cần thiết để lập trình để tiến hành lập trình JTAG (thêm vào các chân nguồn ). Nếu như không cần thiết để đặt điện áp 12v bên ngoài .Cầu chì JTAGEN phải được lập trình và bit JTD trong thanh ghi MCUCSR phải được xóa để kích hoạt cộng truy nhập kiểm tra JTAG

Tính năng lập trình JTAG hỗ trợ :

- Lập trình Flash và quá trình phân tích
- Lập trình EEPROM và quá trình phân tích
- Lập trình cầu chì và quá trình phân tích
- Lập trình bit khóa quá trình phân tích

Bít khóa an toàn thì chính xác như trong chế độ lập trình song song .Nếu như các bít khóa LB1 và LB2 được lập trình ,cầu chì ODCEN không thể được lập trình trừ khi hành động đầu tiên của chip bị xóa .đây là một đặc tính bảo mật mà đảm bảo rằng không có cửa ra back – door cho việc đọc ngoài các thành phần của thiết bị .

Chi tiết trên việc lập trình thông qua giao diện JTAG và các lệnh xác định JTAG được đưa ra trong phần ⑧ Lập trình thông qua giao diện JTAG trang 305.

## Thư mục

Để thêm thông tin chung về chuỗi quét biên ,tham khảo các tiêu chuẩn sau:

### IEEE 1149 (JTAG) quét biên

#### Đặc điểm

- Giao diện JTAG (phù hợp với tiêu chuẩn IEEE 1149)
- Tính năng quét biên theo chuẩn JTAG
- Quét đầy đủ tất cả các cổng chức năng tốt như mạch tương tự có kết nối Off-chip
- Hỗ trợ lựa chọn lệnh IDCODE
- Thêm vào lệnh AVR\_RESET để reset AVR

### Tổng quan hệ thống

Chuỗi quét biên có tính năng của quá trình điều khiển và quan sát các mức logic trên các chân I/O số. tốt như biên giữa cổng logic số và tương tự cho các mạch tương tự có kết nối OFF – chip .Tại mức hệ thống, tất cả IC Có tính năng JTAG được nối tiếp với các tín hiệu TDITDO tới dạng một thanh ghi sip dài .Một bộ điều khiển bên ngoài cài đặt thiết bị để điều khiển giá trị tại các chân đầu ra của chúng ,và quan sát các giá trị đầu vào được nhận từ các thiết bị khác .Bộ điều khiển so sánh các dữ liệu nhận với các kết quả mong muốn .Theo cách này chuỗi quét biên cung cấp một công cụ cho việc kiểm tra các liên kết và các dữ liệu của các thành phần trên bo mạch in bằng cách sử dụng 4 tín hiệu TAP

4 tiêu chuẩn IEEE 1149.1 xác định các lệnh JTAG bắt buộc :IDCODE ,BYPASS,SAMPLE/PRE LOAD,va EXTEST ,như là những lệnh thông dụng xác định AVR \_RESET có thể được sử dụng cho việc kiểm tra bo mạch in .Khởi tạo một quá trình quét của đường dẫn thanh ghi dữ liệu sẽ chỉ ra mã ID của thiết bị .Do mã IDCODE là lệnh mặc định của JTAG .Nó có thể không được xét đến để có một thiết bị được reset trong chế độ kiểm tra .Nếu không reset các đầu vào tới thiết bị có thể bị kết thúc bằng quá trình quét ,và phần mềm bên trong có thể trong một trạng thái không xác định khi thoát ra khỏi chế độ kiểm tra .Nhập lệnh reset ,các đầu vào của bất cứ một chân cổng nào sẽ truy nhập ngay vào trạng thái kháng cao, làm cho lệnh HIGHZ bị thừa .Nếu cần lệnh BYPASS có thể có thể được ban hành để làm cho chuỗi quét ngắn nhất có thể đi qua thiết bị ,thiết bị có thể cài đặt trong trạng thái reset bằng việc kéo chân reset ngoài xuống mực thấp hoặc đưa ra lệnh AVR\_RESET với sự cài đặt của thanh ghi dữ liệu reset

Lệnh EXTEST được sử dụng cho các chân lấy mẫu bên ngoài và các chân tải dữ liệu ra .Dữ liệu từ chổ chốt đầu ra sẽ được điều khiển ra bên ngoài trên các chân ngay khi lệnh EXTEST được tải vào trong thanh ghi IR JTAG .vì vậy các lệnh SAMPLE /PRELOAD cũng nên được sử dụng cho việc cài đặt khởi tạo cho các giá trị đến vòng

quét ,để tránh sự hư hỏng cho bo mạch khi đưa ra lệnh EXTEST trong lần đầu tiên. Các lệnh SAMPLE /PRELOAD cũng có thể được sử dụng để tạo ra một snatshot của các chân bên ngoài trong suốt quá trình hoạt động bình thường của các bộ phận.

Cầu chì JTAGEN phải được lập trình và bít JTD trong thanh ghi I/O MCUCSR phải bị được xóa để kích hoạt để truy nhập kiểm tra JTAG .

Khi sử dụng giao diện JTAG cho việc quét biên ,việc sử dụng một tần số xung nhịp JTAGTCK cao hơn tần số của chip bên trong là có thể được .Xung nhịp của chíp thì không thể chạy .

## Các thanh ghi dữ liệu

Các thanh ghi dữ liệu xác đáng cho quá trình điều khiển quét biên là ;

- Thanh ghi Bypass
- Thanh ghi nhận dạng thiết bị
- Thanh ghi Reset
- Chuỗi quét biên

## Thanh ghi Bypass

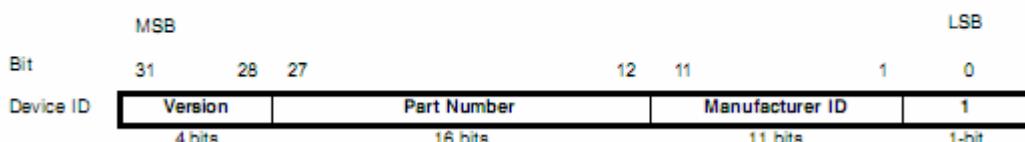
Thanh ghi bypass bao gồm 1 cổng thanh ghi Shift đơn . Khi thanh ghi Bypass được lựa chọn như đường dẫn giữa TDI và TDO , thanh ghi được reset lên 0 khi di chuyển trạng thái bộ điều khiển DR-Capture . Thanh ghi bypass có thể được sử dụng để rút ngắn chuỗi quét trên hệ thống khi mà các thiết bị khác được kiểm tra

## Thanh ghi nhận dạng thiết bị

Hình 122 chỉ ra cấu trúc của thanh ghi nhận dạng thiết bị

Hình 122 : dạng của thanh ghi nhận dạng thiết bị

**Figure 122. The Format of the Device Identification Register**



### Version

Phiên bản (version ) là một số 4-bit nhận dạng kiểm tra các thành phần của phiên bản . số Phiên bản JTAG được kèm theo sự nhận dạng của thiết bị , và được đóng gói tại bit nhận dạng P(0xF) . Nhận dạng A và Q là 0x0 , nhận dạng B và R là 0x1 .

### Phần số thứ tự

Số thứ tự là một mã 16 pít là thành phần nhận dạng

**Table 100.** AVR JTAG Part Number

Part Number	JTAG Part Number (Hex)
ATmega128	0x9702

## Mã sản xuất

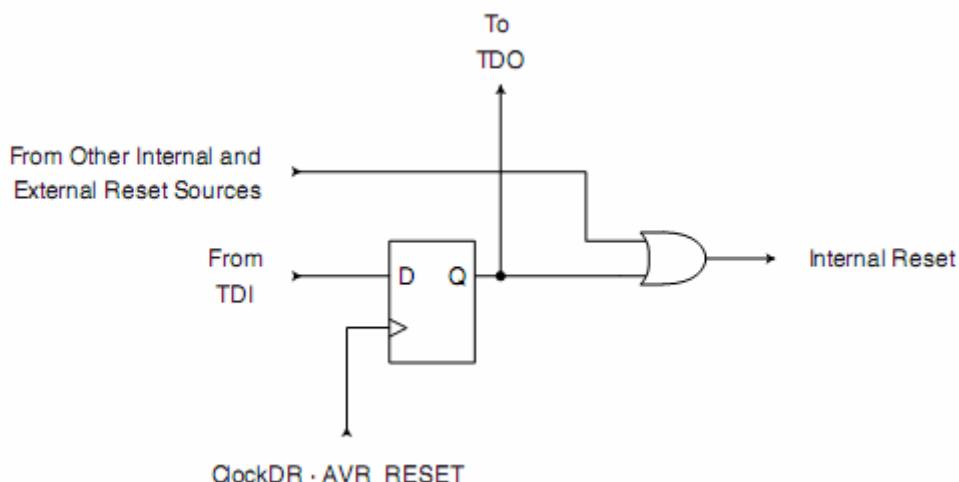
Mã nguồn sản xuất là một mã 11 bit nhận dạng nhà sản xuất ,mã sản xuất của JTAG của ATMEL được liệt kê bảng 101

**Table 101.** Manufacturer ID

Manufacturer	JTAG Manufacturer ID (Hex)
ATMEL	0x01F

Thanh ghi reset là thanh ghi dữ liệu kiểm tra sử dụng để reset các bộ phận do đó AVR có 3 các công 3 trạng thái khi reset ,thanh ghi reset cũng có thể thay đổi chức năng của những lựa chọn JTAG không được lập trình lệnh HIGHZ .

Một giá trị cao trong thanh ghi reset tương ứng để kéo reset ngoài đến mức thấp phần này được reset chỉ khi có một mức giá trị cao được đưa ra trong thanh ghi reset .phụ thuộc vào việc cài đặt bút cầu chì và lựa chọn khóa phần sẽ được reset còn lại trong chu kỳ reset time-out (tham khảo phần nguồn phát sóng nhịp trang 37.sau khi việc giải phóng thanh ghi reset đầu ra từ thanh ghi dữ liệu thì không được chốt, vì vậy reset sẽ xảy ra ngay lập tức ,như là chỉ ra trong hình 123.

**Figure 123.** Reset Register

## Chuỗi quét biên

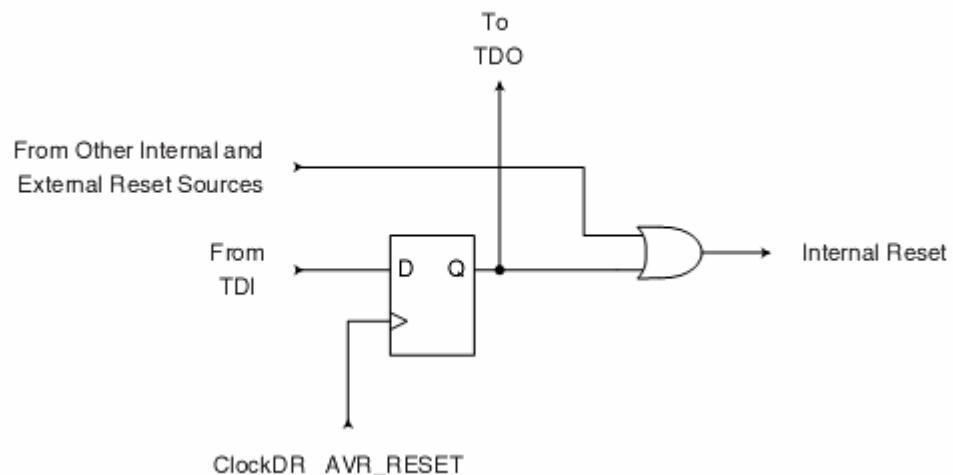
Chuỗi quét biên là tính năng của quá trình điều khiển và phân tích mức lô rích trên các cổng vào ra kỹ thuật số, tốt như biên giữa các khối lô rích tương tự và số cho các mạch tương tự có các kết nối off-chip  
Xem trang 255 để có sự miêu tả chi tiết .

## Các lệnh JTAG xác định các chuỗi quét biên Thanh ghi reset

Thanh ghi reser Là thanh ghi dữ liệu kiểm tra, được sử dụng để reset bộ phận. Do đó các chân cổng 3 trạng thái của AVR khi reset ,thanh ghi reset có thể thay đổi chức năng của các lệnh JTAG lựa chọn không được cài đặt HIGHZ

Một giá trị cao trong thanh ghi reset tương ứng sẽ kéo reset ngoài xuống thấp.Bộ phận này được reset chỉ cần một mức cao được đưa ra trong thanh ghi reset .phụ thuộc vào việc cài đặt cầu chì cho các cài đặt xung nhịp ,phần này sẽ còn lại reset cho một chu kỳ reset time – out (thanh khảo nguồn phát xung nhịp trên trang 37) sau khi giải phóng thanh ghi reset .Đầu ra từ thanh ghi dữ liệu không được chốt ,vì vậy sự reset sẽ xảy ra ngay lập tức như được chỉ ra ở hình 123

Figure 123. Reset Register



## Chuỗi quét biên

Chuỗi quét biên của quá trình điều khiển và giám sát của các mức logic trên các đầu vào ra số , như là biên giữa khối logic tương tự và khối logic số cho các mạch có kết tương tự có các kết nối offchip .

Xem trang 255 để có sự miêu tả chi tiết

## Các lệnh JTAG xác định chuỗi quét biên

Thanh ghi lệnh có độ rộng 4 bit , hỗ trợ 16 lệnh . Được liệt kê bên dưới là các lệnh JTAG hữu dụng cho quá trình điều khiển quét biên . Chú ý rằng lựa chọn lệnh HIGHZ thì không được cài đặt , nhưng tất cả các đầu ra với đặc tính 3 trạng thái , có thể

được cài đặt trong trạng thái có trở kháng cao bằng việc cài đặt lệnh AVR\_RESET , do đó trạng thái khởi tạo cho tất cả các chân cổng là 3 trạng thái .

Như đã được định nghĩa trong tài liệu này LSB được di chuyển ra ngoài và vào trong lần đầu tiên cho tất cả các thanh ghi SHIFT

OPCODE cho mỗi lệnh được chỉ ra紧跟 sau của mỗi tên lệnh trong định dạng HEX . Đoạn văn bản miêu tả cái mà thanh ghi dữ liệu được lựa chọn như là đường dẫn giữa TDI và TD0 cho mỗi lệnh .

### **EXTEST , \$0**

Các lệnh JTAG bắt buộc cho việc lựa chọn chuỗi quét biên như là thanh ghi dữ liệu cho việc kiểm tra các mạch bên ngoài đến các chân của AVR . Về các chân cổng , quá trình vô hiệu hóa pull\_up , bộ điều khiển đầu ra , dữ liệu đầu ra , dữ liệu đầu vào đều được truy nhập thông qua các chuỗi quét . Mạch tương tự có các kết nối offchip , các dao điện giữa các khối logic số và logic tương tự trong chuỗi quét . Thành phần của các đầu ra đã dc chốt trong các chuỗi quét được điều khiển như là thanh ghi JTAG\_IR được tải với lệnh EXTEST .

Các trạng thái hoạt động là :

- Capture\_DR : dữ liệu trên các chân bên ngoài được lấy mẫu vào trong chuỗi quét biên .
- SHIFT\_DR : Chuỗi quét bên trong được di chuyển bởi đầu vào TCK
- UPDATE\_DR : Dữ liệu từ chuỗi quét được đặt lên các chân đầu ra

### **IDCODE ; \$1**

Lựa chọn lệnh JTAG , sẽ lựa chọn thanh ghi ID 32 bit như là thanh ghi dữ liệu . Thanh ghi ID bao gồm 1 số thứ tự phiên bản , số thứ tự thiết bị và mã nguồn sản xuất được lựa chọn bởi JEDEC . Điều này được mặc định bằng lệnh sau khi bật nguồn .

Các trạng thái là :

- Capture\_DR : Dữ liệu trong thanh ghi IDCODE được lấy mẫu vào trong chuỗi quét
- SHIFT\_DR : Chuỗi quét IDCODE được di chuyển bằng đầu vào TCK

### **SAMPLE\_PRELOTAD : \$2**

Các lệnh JTAG bắt buộc cho việc tải trước các địa chỉ chốt và tao ra 1 Snap\_Shot của các chân đầu vào \ đầu ra thiêu sự ảnh hưởng của quá trình điều khiển hệ thống . Tuy nhiên các đầu vào được chốt thì được nối với các chân . Chuỗi quét được lựa chọn như là thanh ghi dữ liệu

Các trạng thái hoạt động :

- Capture\_DR : Dữ liệu trên các chân bên ngoài được lấy mẫu vào trong chuỗi quét
- SHIFT\_DR : Chuỗi quét bên trong được di chuyển bởi đầu vào TCK
- UPDATE\_DR : Dữ liệu từ chuỗi quét biến được đặt vào trong các đầu ra đã chốt . Tuy nhiên , các đầu ra đã chốt được kết nối với các chân

### **AVR\_RESET ;\$C**

Lệnh JTAG xác định chung cho việc cưỡng bức thiếp bị AVR vào trong chế độ RESET hoặc giải phóng nguồn RESET\_JTAG . Bộ điều khiển TAP thì không được RESET bởi lệnh này. Một bít của thanh ghi RESET được lựa chọn như là thanh ghi dữ liệu . Chú ý rằng RESET sẽ được kích hoạt chỉ cần có mức Logic 0 trong chuỗi RESET . Đầu ra từ chuỗi này thì không được chốt

Các hoạt động “

SHIFT\_DR : Thanh ghi RESET được di chuyển bởi đầu vào TCK

### **BYPASS ;\$F**

Lệnh JTAG bắt buộc đang lựa chọn thanh ghi BYPASS cho thanh ghi dữ liệu

Các hoạt động là :

- Capture\_DR : Tải mức logic 0 vào thanh ghi BYPASS
- SHIFT\_DR : Thanh ghi BYPASS được ngăn giữa các bít TDI và TDO đã được di chuyển

Chuỗi quét biến liên quan đến thanh ghi , trong vùng nhớ I/O

### **Thanh ghi trạng thái mà điều khiển MCU\_MCUCSR :**

Thanh ghi điều khiển và trạng thái MCU : bao gồm các bít điều khiển cho các chức năng MCU chung và cung cấp thông tin đến nguồn RESET gây ra việc RESET MCU .

Bit	7	6	5	4	3	2	1	0	MCUCSR
Read/Write	JTD	-	-	JTRF	WDRF	BORF	EXTRF	PORF	
Initial Value	R/W	R	R	R/W	R/W	R/W	R/W	R/W	See Bit Description

Bít 7\_JTD : Vô hiệu hóa giao diện JTAG

Khi bít này là 0 , giao diện JTAG được kích hoạt nếu như cầu chì JTAGEN được lập trình . Nếu bít này là 1 giao diện JTAG bị vô hiệu hóa . Để tránh các sự kích hoạt hoặc vô hiệu hóa không mong muốn của giao diện JTAG , 1 chuỗi thời gian được lập trình phải được kèm theo khi thay đổi các bít này . Phần mềm ứng dụng phía viết bít này lên 2 lần giá trị được sét đến trong vòng 4 chu kì xung nhịp để thay đổi giá trị của nó .

Nếu giao diện JTAG được gỡ bỏ không kết nối với các mạch JTAG khác , bít JTD lên được cài đặt là 1 . Nguyên nhân của điều này là để tránh trạng thái hiện hành trên chân TDO trong giao diện JTAG

#### Bít 4\_JTRF : Cờ báo RESET JTAG

Bít này được cài đặt nếu 1 sự RESET đang được gây ra bởi mức logic 1 trên thanh ghi RESET JTAG được lựa chọn bởi lệnh JTAG (AVR\_RESET ) . Bít này được RESET bởi chế độ RESET power\_on , hoặc bằng việc viết mức logic 0 lên cờ .

#### **Chuỗi quét biên :**

Chuỗi quét biên có khả năng điều khiển và giám sát . Các mức logic trên các cổng vào ra số như là biên giữa các khối logic số và tương tự cho mạch tương tự có 1 kết nối offchip .

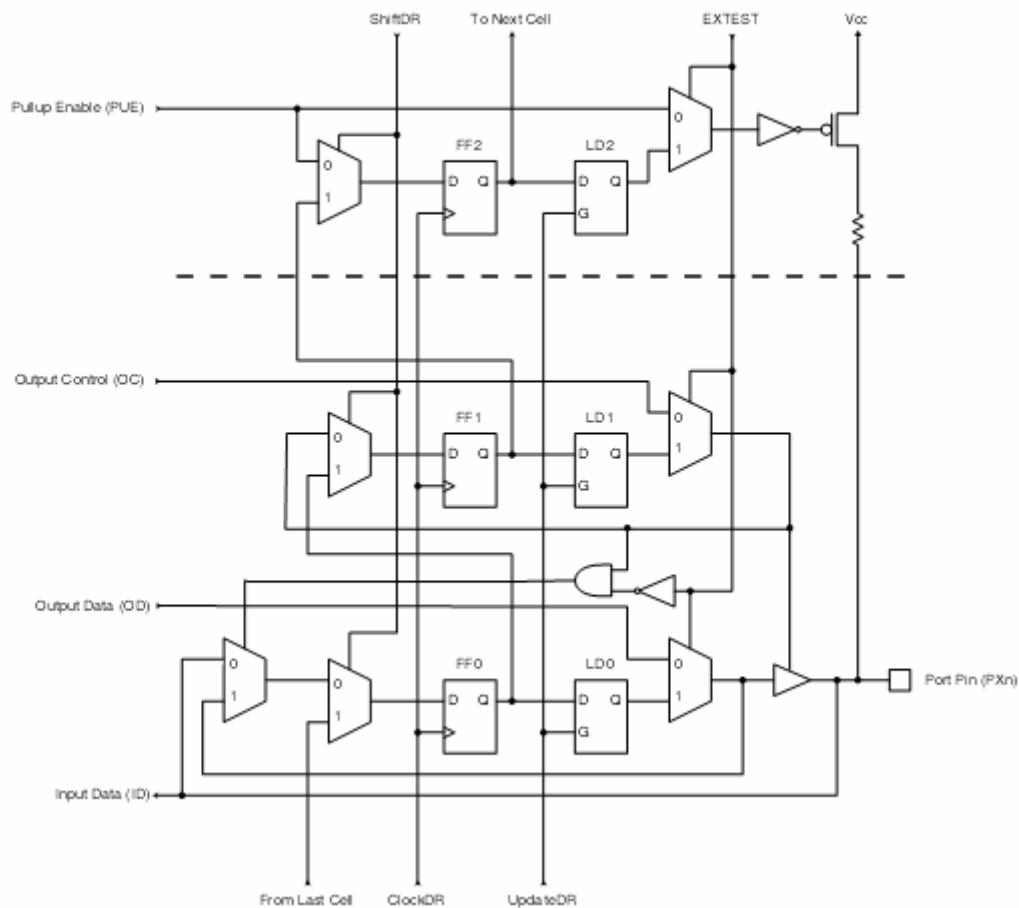
#### **Việc quét các cổng số**

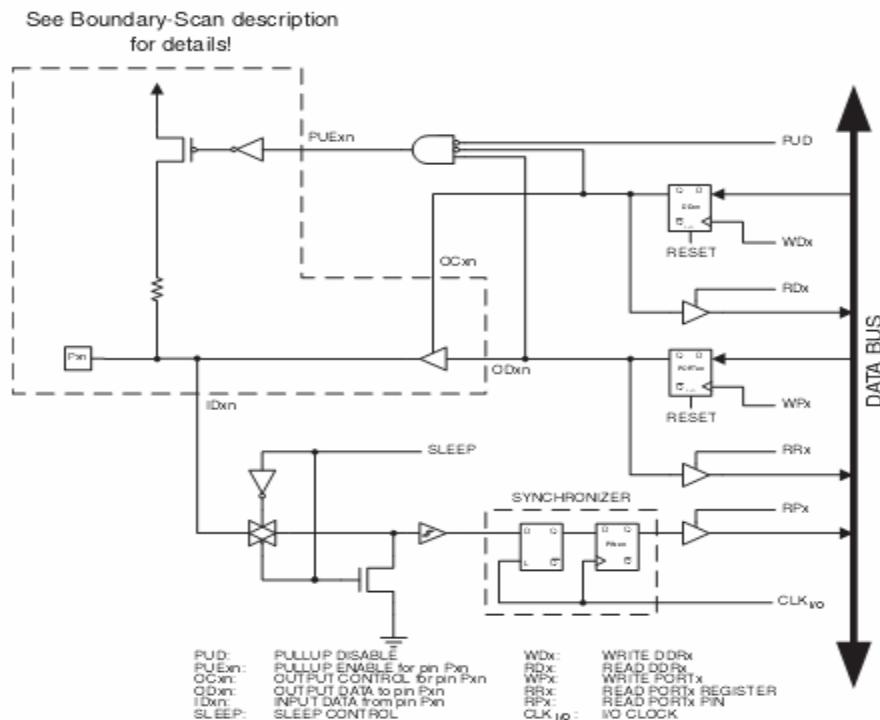
Hình 124 chỉ ra rằng ngăn quét biên cho một cổng hai hướng với chức năng pull – up. Ngăn một ngăn thì bao gồm một ngăn quét biên tiêu chuẩn cho pull – up Enable – PUExn – Function , và một ngăn chân õ hướng mà bao gồm 3 tín hiệu đầu ra điều khiển – OCxn , dữ liệu đầu ra – Odxn , dữ liệu đầu vào – IDxn , vào trong chỉ một thanh ghi shift 2 cổng .các chân và các cổng indexef thì không được sử dụng trong sự miêu ta dưới đây

Khối logic quét biên thì không được vẽ trong hình vẽ tài liệu này .Hình 125 chỉ ra một chân cổng số đơn giản như được miêu tả trong phần các cổng vào rat trang 66. chi tiết của chuỗi quét biên từ hình 124 được thay thế bằng các khôi hộp trong hinh 125.

Khi không có cổng luôn phiên chức năng nào được đưa ra ,đầu vào dữ liệu –ID tương ứng tới giá trị của thanh ghi PINxn (nhưng ID không có bộ đồng bộ hóa)đầu ra dữ liệu tương ứng tới thanh ghi PORT ,đầu ra điều khiển tương ứng tới thanh ghi dữ liệu – DD register,và pull – up Enable – PUExn – Function tương ứng tới các khôi lôgic PUD .DDxn .PORTxn .

Chức năng cổng luôn phiên số được nối bên ngoài hộp DOTTED trong hình 125 để đọc chuỗi quét các giá trị thực sự của chân .về chức năng tương tự ,có sự kết nối trực tiếp của các chân bên ngoài tới các mạch tương tự ,và một chuỗi quet được chen vào giữa dao điện của khôi logic của mạch tương tự

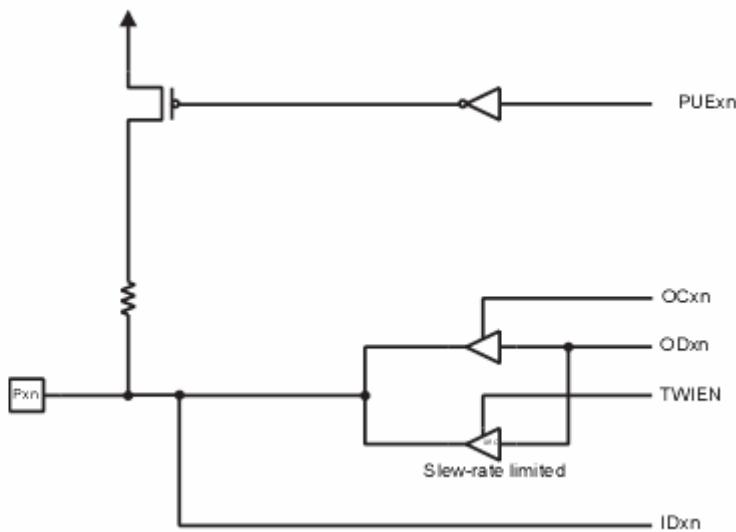
**Figure 124.** Boundary-scan Cell for Bi-directional Port Pin with Pull-Up Function.

**Figure 125.** General Port Pin Schematic diagram

### Chuỗi quét biên và giao diện 2 dây tuân tu

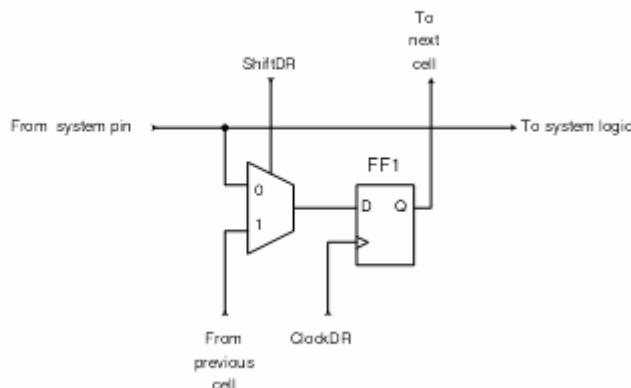
2 chân giao diện tuân tự ô giây SCLSCA có một tín hiệu điều khiển được thêm vào trong chuỗi quét ,kích hoạt giao diện ô dây tuân tự - TWIEN. Như được chỉ ra ở 126 tín hiệu TWIEN kích hoạt một bộ đếm 3 trạng thái với bộ điều khiển slew- rate trong chế độ song song với các chân cổng số chính thức .một chuỗi quét chung được chỉ ra ở trên hình 130 thì được tận công bởi tín hiệu twien

- Notes:
1. A separate scan chain for the 50 ns spike filter on the input is not provided. The ordinary scan support for digital port pins suffice for connectivity tests. The only reason for having TWIEN in the scan path, is to be able to disconnect the slew-rate control buffer when doing boundary-scan.
  2. Make sure the OC and TWIEN signals are not asserted simultaneously, as this will lead to drive contention.

**Figure 126.** Additional Scan Signal for the Two-wire Interface

### Việc quét các chân reset

Chân reset chấp nhận điện áp 5v ổn định cho quá trình điều khiển reset tiêu chuẩn ,mức logic cao hoạt động 12 v cho quá trình lập trình song song điện áp cao .một sự giám sát được chỉ ra trong hình 127 được chen vào giữa chân tín hiệu reset 5v-RSTT và tín hiệu reset 12v-RSTHV .

**Figure 127.** Observe-only Cell

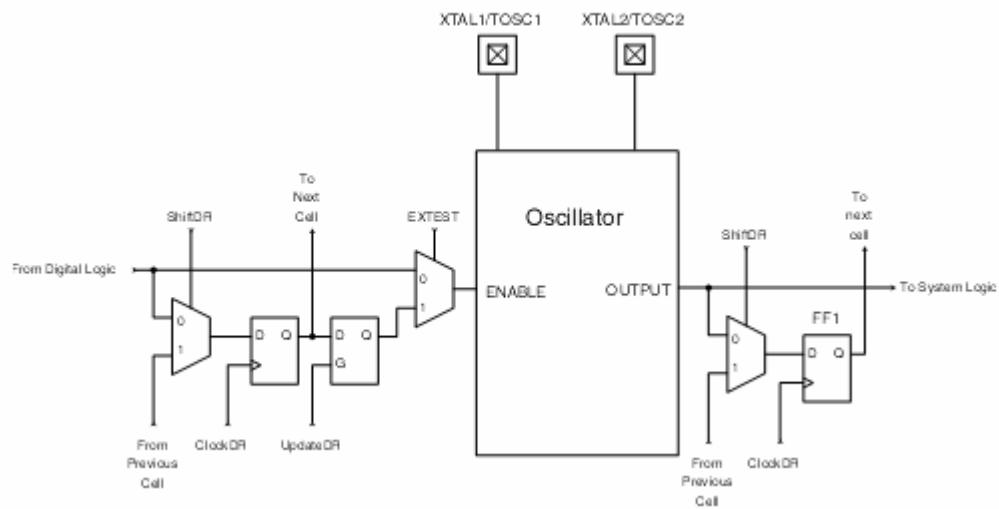
### Quét các chân khóa

Các thiết bị AVR có nhiều sự lựa chọn chân khóa có thể lựa chọn bằng các cầu chì .Đó là bộ tạo dao động RC bên trong ,bộ tạo dao động RC bên ngoài ,bộ tạo dao động thạch anh tần số cao ,bộ tạo dao động tần số thấp ,và bộ khuỷch đại gồm .

Hình 128 chỉ ra cách mà mỗi tổ tạo dao động với các kết nối ngoài được hỗ trợ trong chuỗi quét .tín hiệu kích hoạt được hỗ trợ bởi chuỗi quét biên chung ,trong khi bộ tạo giao động /xung nhịp bên ngoài bị tấn công tới một ngăn quan sát .thêm vào đó xung nhịp chính bộ tạo dao động timer được quét bằng cách

giống nhau .đầu ra từ bộ tạo dao động RC bên trong không được quét, như là bộ tạo dao động này không được kết nối với bên ngoài

**Figure 128.** Boundary-scan Cells for Oscillators and Clock Options



Bảng 102 liệt kê các thanh ghi quét cho chân tạo xung nhịp bên ngoài XTAL1 ,bộ tạo dao động với các kết nối XTAL1/XTAL2 như là bộ tạo dao động Timer 32 kHz.

**Table 102.** Scan Signals for the Oscillators<sup>(1)(2)(3)</sup>

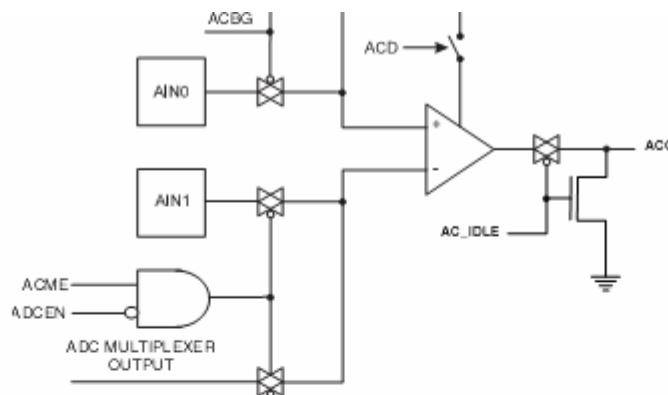
Enable signal	Scanned Clock Line	Clock Option	Scanned Clock Line when not Used
EXTCLKEN	EXTCLK (XTAL1)	External Clock	0
OSCON	OSCCK	External Crystal External Ceramic Resonator	0
RCOSCEN	RCCK	External RC	1
OSC32EN	OSC32CK	Low Freq. External Crystal	0
TOSKON	TOSCK	32 kHz Timer Oscillator	0

- Notes:
1. Do not enable more than one clock source as main clock at a time.
  2. Scanning an Oscillator output gives unpredictable results as there is a frequency drift between the Internal Oscillator and the JTAG TCK clock. If possible, scanning an external clock is preferred.
  3. The clock configuration is programmed by fuses. As a fuse is not changed run-time, the clock configuration is considered fixed for a given application. The user is advised to scan the same clock option as to be used in the final system. The enable signals are supported in the scan chain because the system logic can disable clock options in sleep modes, thereby disconnecting the Oscillator pins from the scan path if not provided. The INTCAP fuses are not supported in the scan-chain, so the boundary scan chain can not make a XTAL Oscillator requiring internal capacitors to run unless the fuse is correctly programmed.

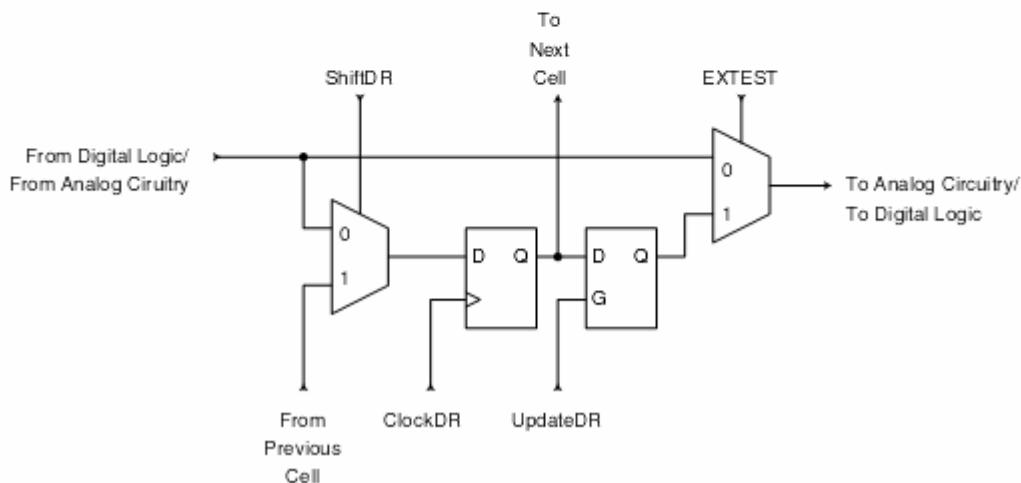
## Quét bộ so sánh tương tự

Tín hiệu bộ so sánh thích đáng bắt chấp chuỗi quét được chỉ ra trong hình 129. ngăn chuỗi quét từ hình 130 bị tấn công bởi mỗi một trong các tín hiệu này . Các tín hiệu được miêu tả trong bảng 103.

Bộ so sánh không cần phải sử dụng cho việc kiểm tra kết nối, do đó tất cả các đầu vào tương tự bị chia sẻ với một cổng đầu ra số



**Figure 130.** General Boundary-scan Cell used for Signals for Comparator and ADC

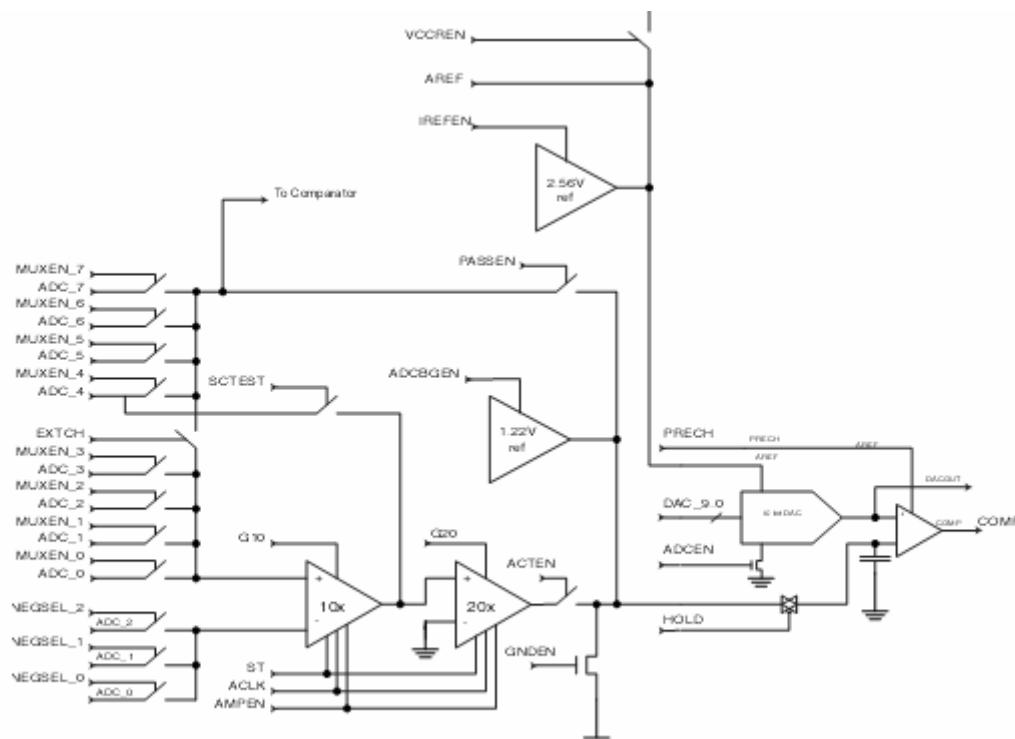


**Table 103.** Boundary-scan Signals for the Analog Comparator

Signal Name	Direction as Seen from the Comparator	Description	Recommended Input when not in Use	Output values when Recommended Inputs are Used
AC_IDLE	Input	Turns off Analog comparator when true	1	Depends upon µC code being executed
ACO	Output	Analog Comparator Output	Will become input to µC code being executed	0
ACME	Input	Uses output signal from ADC mux when true	0	Depends upon µC code being executed
ACBG	Input	Bandgap Reference enable	0	Depends upon µC code being executed

## Quét ADC

Hình 131 chỉ ra một sơ đồ khối của khối ADC với các tín hiệu quan sát và điều khiển thích đáng .Ngăn quét biên từ hình 127 bị tần công tới mỗi một trong các tín hiệu .Khối ADC không cần sử dụng sự kiểm tra kết nối ,do đó tất cả các đầu vào tương tự bị chia sẽ với một chân cổng số



**Table 104.** Boundary-scan Signals for the ADC

Signal Name	Direction as Seen from the ADC	Description	Recommended Input when not in Use	Output Values when Recommended Inputs are Used, and CPU is not Using the ADC
COMP	Output	Comparator Output	0	0
ACLK	Input	Clock signal to gain stages implemented as Switch-cap filters	0	0
ACTEN	Input	Enable path from gain stages to the comparator	0	0
ADCBGEN	Input	Enable Band-gap reference as negative input to comparator	0	0
ADCEN	Input	Power-on signal to the ADC	0	0
AMPEN	Input	Power-on signal to the gain stages	0	0
DAC_9	Input	Bit 9 of digital value to DAC	1	1
DAC_8	Input	Bit 8 of digital value to DAC	0	0
DAC_7	Input	Bit 7 of digital value to DAC	0	0
DAC_6	Input	Bit 6 of digital value to DAC	0	0
DAC_5	Input	Bit 5 of digital value to DAC	0	0
DAC_4	Input	Bit 4 of digital value to DAC	0	0
DAC_3	Input	Bit 3 of digital value to DAC	0	0
DAC_2	Input	Bit 2 of digital value to DAC	0	0
DAC_1	Input	Bit 1 of digital value to DAC	0	0
DAC_0	Input	Bit 0 of digital value to DAC	0	0
EXTCH	Input	Connect ADC channels 0 - 3 to bypass path around gain stages	1	1
G10	Input	Enable 10x gain	0	0
G20	Input	Enable 20x gain	0	0

Table 104. Boundary-scan Signals for the ADC (Continued)

Signal Name	Direction as Seen from the ADC	Description	Recommended Input when not in Use	Output Values when Recommended Inputs are Used, and CPU is not Using the ADC
GNDEN	Input	Ground the negative input to comparator when true	0	0
HOLD	Input	Sample & Hold signal. Sample analog signal when low. Hold signal when high. If gain stages are used, this signal must go active when ACLK is high.	1	1
IREFEN	Input	Enables Band-gap reference as AREF signal to DAC	0	0
MUXEN_7	Input	Input Mux bit 7	0	0
MUXEN_6	Input	Input Mux bit 6	0	0
MUXEN_5	Input	Input Mux bit 5	0	0
MUXEN_4	Input	Input Mux bit 4	0	0
MUXEN_3	Input	Input Mux bit 3	0	0
MUXEN_2	Input	Input Mux bit 2	0	0
MUXEN_1	Input	Input Mux bit 1	0	0
MUXEN_0	Input	Input Mux bit 0	1	1
NEGSEL_2	Input	Input Mux for negative input for differential signal, bit 2	0	0
NEGSEL_1	Input	Input Mux for negative input for differential signal, bit 1	0	0
NEGSEL_0	Input	Input Mux for negative input for differential signal, bit 0	0	0
PASSEN	Input	Enable pass-gate of gain stages.	1	1
PRECH	Input	Precharge output latch of comparator. (Active low)	1	1

Table 104. Boundary-scan Signals for the ADC (Continued)

Signal Name	Direction as Seen from the ADC	Description	Recommended Input when not in Use	Output Values when Recommended Inputs are Used, and CPU is not Using the ADC
SCTEST	Input	Switch-cap TEST enable. Output from x10 gain stage send out to Port Pin having ADC_4	0	0
ST	Input	Output of gain stages will settle faster if this signal is high first two ACLK periods after AMPEN goes high.	0	0
VCCREN	Input	Selects Vcc as the ACC reference voltage.	0	0

Note: Incorrect setting of the switches in Figure 131 will make signal contention and may damage the part. There are several input choices to the S&H circuitry on the negative input of the output comparator in Figure 131. Make sure only one path is selected from either one ADC pin, Bandgap reference source, or Ground.

Nếu bộ ADC không được sử dụng trong quá trình quét các giá trị đầu vào để nghị từ bảng 104 nên được sử dụng .Người sử dụng được khuyến cáo rằng không sử dụng các cổng khuyếch đại khác nhau trong suốt quá trình quét .chuyển mạch trên cơ sở cổng khuyếch đại cần thiết hoạt động nhanh và thời gian chính xác mà khó để quan sát khi sử dụng một chuỗi quét chi tiết của việc điều khiển của các cổng khuyếch đại khác nhau vì vậy không được cung cấp .

Bộ ADC được xây dựng trên mạch tương tự như được chỉ trong hình 131 với một thuật toán lập trình liên tiếp được cài đặt trong khối logic số .khi sử dụng một chuỗi quét biên ,vấn đề thì thường được đảm bảo đặt một điểm áp tương tự được đo trong vòng một vài giới hạn .điều này có thể dễ dàng được thực hiện mà thiếu việc chạy một thuật toán liên tiếp :áp dụng cho dưới hạn ký thuật số DAC [9:0], đảm bảo rằng từ đầu từ bộ so sánh ở mức thấp sau đó đặt giới hạn cao hơn ký thuật số [9:0],và kiểm tra đầu ra từ bộ so sánh ở mức cao.

Bộ ADC Không cần sử dụng kiểm tra kết nối ,do đó tất cả các đầu ra bị chia sẻ với các cổng ra số

Khi sử dụng bộ ADC cần nhớ các điểm sau :

- Chân công cho kênh ADC đang sử dụng phải được cấu hình là một đầu ra với chức năng pull-up bị vô hiệu hóa để tránh các tín hiệu không mong muốn

- Trong chế độ bình thường ,một quá trình chuyển đổi dummy (bao gồm 10 biến so sánh )được tiến hành khi đang kích hoạt ADC .Người sử dụng được khuyên rằng nên đợi trong khoảng thời gian dưới 200ns sau khi việc kích hoạt ADC trước khi điều khiển /quan sát bất cứ tín hiệu ADC nào ,hoặc tiến hành một quá trình chuyển đổi dummy trước khi sử dụng kết quả đầu tiên

- Giá trị DAC phải ổn định tại giá trị điểm giữa 0 và 200 khi đang có tín hiệu HOLD thấp (chế độ mẫu) như là một ví dụ ,coi như nhiệm vụ của việc phân tích một tín hiệu đầu vào 1.5v cộng trừ 5 % tại kênh ADC kênh số 3 khi điện áp nguồn cấp là 5v và ARFF được kết nối ra ngoài kênh VCC

Giá trị được đề nghị từ bảng 104 thì được sử dụng trừ khi các giá trị khác đưa ra trong bảng 105 .Chỉ có các giá trị chân cổng DAC của chuỗi quét được chỉ ra .Cột actionf miêu tả lệnh jtag được sử dụng trước khi điền đầy vào thanh ghi quét biên với các cột liên tiếp . Quá trình kiểm tra nên được thực hiện trên chuỗi khi đang quét dữ liệu khi dữ liệu trên các dòng giống nhau trong bảng .

**Table 105.** Algorithm for Using the ADC

Step	Actions	ADCEN	DAC	MUXEN	HOLD	PRECH	PA3. Data	PA3. Control	PA3. Pullup_ Enable
1	SAMPLE_PRELOAD	1	0x200	0x08	1	1	0	0	0
2	EXTEST	1	0x200	0x08	0	1	0	0	0
3		1	0x200	0x08	1	1	0	0	0
4		1	0x123	0x08	1	1	0	0	0
5		1	0x123	0x08	1	0	0	0	0
6	Verify the COMP bit scanned out to be 0	1	0x200	0x08	1	1	0	0	0
7		1	0x200	0x08	0	1	0	0	0
8		1	0x200	0x08	1	1	0	0	0
9		1	0x143	0x08	1	1	0	0	0
10		1	0x143	0x08	1	0	0	0	0
11	Verify the COMP bit scanned out to be 1	1	0x200	0x08	1	1	0	0	0

Sử dụng thuật toán thời gian bao gồm tín hiệu HOLD ép buộc tần số xung nhịp TCK . Như là một thuật toán giữ HOLD ở mức cao cho 5 bước cho năm bước, tần số xung nhịp TCK dưới năm lần của chuỗi quét.

## Thứ tự chuỗi quét của atmega 128

Bảng 106 chỉ ra thứ tự chuỗi quét của TDI và TDO khi chuỗi quét được lựa chọn như là một đường dẫn dữ liệu. Bit 0 trong LSB; bit đầu tiên được quét; và bit cuối cùng được quét ra ngoài. Thứ tự quét kèm theo thứ tự chân bên ngoài là có thể. Vì vậy các bit của cổng A được quét trong bit đối diện với các cổng khác. Ngoại trừ các nguyên tắc của chuỗi quét cho các mạch tương tự, cái mà cấu tạo nên các bit có trọng số lớn nhất của chuỗi quét bắt chấp việc các chân vật lý của chúng được kết nối. Trong hình 124 PXn tương ứng với FF0, PXn điều khiển tương ứng FF1, PXn. Kích hoạt PULLUP tương ứng với FF1. Bit 2, 3, 4, 5 của cổng C thì không ở trong chuỗi quét, do đó các chân này cấu thành nên các chân TAP khi JTAG được kích hoạt

**Table 106.** ATmega128 Boundary-scan Order

Bit Number	Signal Name	Module
204	AC_IDLE	Comparator
203	ACO	
202	ACME	
201	AINBG	
200	COMP	ADC
199	PRIVATE_SIGNAL1 <sup>(1)</sup>	
198	ACLK	
197	ACTEN	
196	PRIVATE_SIGNAL1 <sup>(2)</sup>	
195	ADCBGEN	
194	ADCEN	
193	AMPEN	
192	DAC_9	
191	DAC_8	
190	DAC_7	
189	DAC_6	
188	DAC_5	
187	DAC_4	
186	DAC_3	
185	DAC_2	
184	DAC_1	
183	DAC_0	
182	EXTCH	
181	G10	
180	G20	
179	GNDEN	
178	HOLD	
177	IREFEN	
176	MUXEN_7	

## XXIII . Hỗ trợ bộ tải chương trình mồi lập trình read – while – write

Hỗ trợ tải chương trình mồi cung cấp 1 cơ cấu lập trình cho việc cập nhật và nạp lên đoạn mã chương trình bằng chính MCU . Đặc điểm này cho phép ứng dụng các phần mềm ứng dụng cập nhật được điều khiển bởi MCU bằng cách sử dụng một chương trình tải chương trình boot lưu trú trong bộ nhớ Flash . Chương trình tải boot có thể sử dụng bất cứ bất cứ giao diện dữ liệu khả dụng nào và giao thức liên kết nào để đọc mã và viết chương trình ở bên trong bộ nhớ Flash , hoặc đọc code từ

### Đặc điểm của bộ khởi động tải

- Read-While-Write Self-Programming
- Kích cỡ bộ nhớ khởi động linh hoạt.
- Tính bảo mật cao ( phân chia các bít khóa khởi động cho một sự bảo vệ linh hoạt)
- Phân chia các bit cần chì đến các vector lựa chọn reset
- Tối ưu hóa kích cỡ trang
- Thuật toán mã thuận tiện
- Hỗ trợ Efficient Read-Modify-Write

Chú ý: một trang trong phần này trong bộ nhớ flash bao gồm nhiều byte ( xem bảng 123 trang 291 ). Được sử dụng trong suốt quá trình lập trình.

### Ứng dụng và khởi động bộ tải vùng dữ liệu flash

Bộ nhớ flash được tổ chức thành hai phần chính, khu vực ứng dụng và khu vực tải khởi động (xem hình 133). Kích cỡ của các vùng khác nhau được cấu hình bởi cấu chì BOOTSZ được chỉ ra trong trang 284 và hình 133. Có hai vùng nhớ có thể có các mức khác nhau của chế độ bảo vệ do đó chúng có các sự cài đặt khác nhau cho các bit khóa

### Khu vực ứng dụng

Khu vực ứng dụng là khu vực của bộ nhớ flash được sử dụng để lưu trữ các mã ứng dụng. Mức bảo vệ cho khu vực ứng dụng có thể được lựa chọn bằng các bit khóa khởi động ứng dụng ( các bít khóa khởi động 0 ) xem bảng 275. Khu vực ứng dụng có thể không bao giờ lưu trữ bất cứ mã khởi động tải nào từ lệnh SPM bị vô hiệu hóa khi thực thi từ vùng ứng dụng.

## Vùng khởi động bộ tải-BLS

Trong khi khu vực ứng dụng được sử dụng để lưu trữ mã ứng dụng, phần mềm khởi động bộ tải phải được đặt trong BLS do đó lệnh SPM có thể khởi tạo một chương trình khi đang thực thi từ BLS. Lệnh SPM có thể truy cập toàn bộ bộ nhớ flash, bao gồm chính bản thân BLS mức bảo vệ cho khu vực khởi động tải được lựa chọn bằng các bit khóa khởi động bộ tải (bit khóa khởi động 1) bảng 109 trang 276.

## Read-While-Write and No Read-While-Write Flash Sections

CPU hỗ trợ đặt trong khi viết hoặc nếu CPU bị dừng trong suốt quá trình cập nhật phần mềm khởi động bộ tải thì phụ thuộc vào địa chỉ cái mà đang được lập trình.Thêm vào đó hai khu vực mà được cấu hình bởi cầu chì BOOTSZ như là được miêu tả ở bên trên, bộ nhớ flash được chia vào trong 2 khu vực ổn định khu vực đọc trong khi viết (RWW) và khu vực không đọc trong khi viết (NRWW). Giới hạn giữa RWW và (NRWW) được đưa ra trong bảng.

Chú ý: trên trang 284 và hình 133 trang 275. Sự khác nhau chính giữa hai khu vực này là.

- Khi xóa hoặc viết một trang được đặt trong khu vực RWW, khu vực NRWW có thể đọc suốt trong quá trình điều khiển.
- Khi xóa hoặc viết một trang được đặt trong khu vực NRWW, CPU bị dừng trong toàn bộ quá trình điều khiển

Chú ý rằng người sử dụng phần mềm có thể không bao giờ đọc bất cứ mã nào được đặt bên trong khu vực RWW trong suốt quá trình hoạt động của phần mềm khởi động bộ tải. Cú pháp “Read-While Write section” tham chiếu tới khu vực mà đang được lập trình (được xóa hoặc được viết) không phải khu vực mà thực sự đang được đọc trong suốt quá trình cập nhật khởi động bộ tải phần mềm.

## Khu vực Read-While Write section – RWW

Nếu một quá trình cập nhật khởi động bộ tải phần mềm đang lập trình một trang bên trong khu vực RWW. Nó có thể đọc mã từ bộ nhớ flash, nhưng chỉ có một mã đặt trong khu vực NRWW. Trong suốt 1 sự lập trình đang tiến hành, phần mềm phải đảm bảo rằng khu vực RWW đang được đọc. Nếu người sử dụng phần mềm đang cố gắng đọc mã mà đặt bên trong khu vực RWW (ví dụ bằng các lệnh call/ jmp/lpm hoặc một ngắt) trong suốt sự lập trình, phần mềm có thể kết thúc trong một trạng thái không xác định. Để tránh điều này, các ngắt nên được vô hiệu hóa hoặc di chuyển tới khu vực khởi động bộ tải. Khu vực khởi động bộ tải thì luôn đặt trong NRWW. Bit báo bận khu vực RWW trong thanh ghi trạng thái và điều khiển trạng thái bộ nhớ chương trình lưu trữ (SPMCSR) sẽ được đọc như là một logic 1 chỉ cần khu vực RWW bị khóa khỏi việc đọc. Sau một sự lập trình được hoàn thành, RWWSB phải bị xóa bằng phần mềm trước khi đọc mã đặt trong khu vực RWW. Xem thanh ghi trạng thái và điều khiển trạng thái bộ

nhớ chương trình lưu trữ (SPMCSR) trang 277. Để biết thêm chi tiết về cách xóa RWWSB

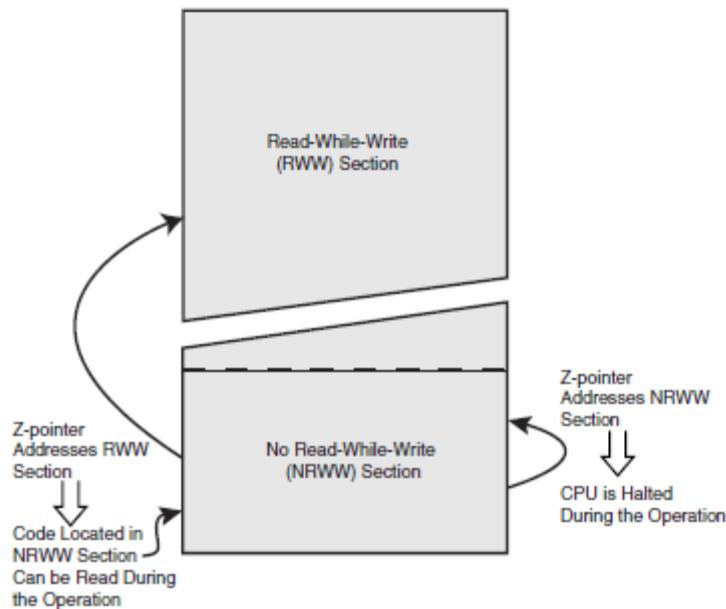
### No Read-Whlie-Write Section – NRWW

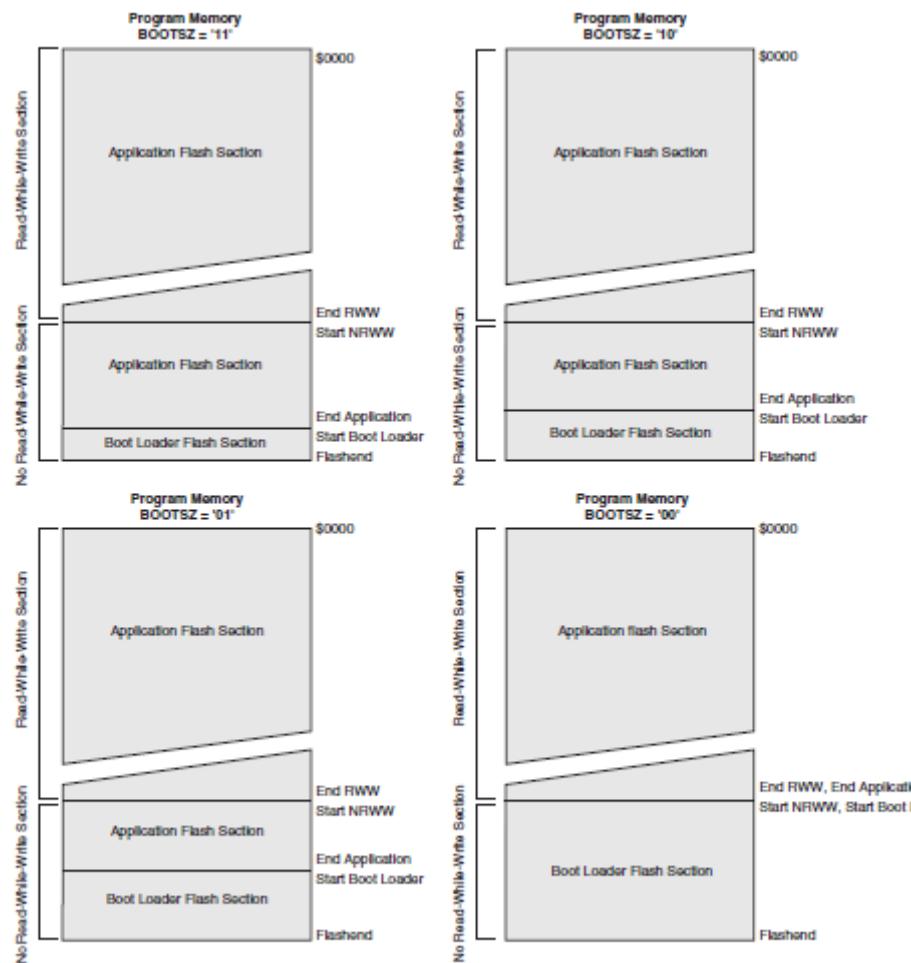
Đoạn mã đặt trong khu vực NRWW có thể được đọc khi phần mềm khởi động bộ tải đang cập nhật một trang trong khu vực RWW. Khi mã khởi động bộ tải cập nhật khu vực RNWW, CPU bị dừng trong suốt quá trình bị viết hoặc xóa một trang.

**Table 107. Read-While-Write Features**

Which Section does the Z-pointer Address During the Programming?	Which Section can be Read During Programming?	Is the CPU Halted?	Read-While-Write Supported?
RWW section	NRWW section	No	Yes
NRWW section	None	Yes	No

**Figure 132. Read-While-Write vs. No Read-While-Write**



**Figure 133. Memory Sections<sup>(1)</sup>**

Note: 1. The parameters in the figure above are given in [Table](#) on page 284.

## Các bit khóa khởi động bộ tải

Nếu tính năng khởi động bộ tải là cần thiết, toàn bộ bộ nhớ flash là hữu dụng cho mã ứng dụng. Khởi động bộ tải có hai sự cài đặt riêng biệt của các bit khóa khởi động mà có thể được cài đặt một cách độc lập. Điều này đưa người sử dụng một sự lựa chọn linh hoạt để lựa chọn các mức khác nhau của quá trình bảo vệ.

- Người sử dụng có thể lựa chọn
- Để bảo vệ toàn bộ bộ nhớ flash khỏi một sự cập nhật phần mềm bằng MCU
- Để bảo vệ chỉ nguyên khu vực khởi động bộ tải flash khỏi một sự cập nhật phần mềm bằng MCU
- Để bảo vệ chỉ khu vực ứng dụng flash khỏi một sự cập nhật phần mềm bằng MCU
- Cho phép cập nhật phần mềm cho bộ nhớ flash

Xem bảng 108 và 109 để biết thêm thông tin chi tiết. Các bit khóa khởi động có thể được cài đặt trong phần mềm và trong chế độ lập trình song song, nối tiếp, nhưng chúng có thể bị xóa bởi một lệnh xóa chíp. Bit khóa ghi chung (bit khóa chế độ hai) không điều khiển sự lập trình của bộ nhớ flash bằng lệnh SPM. Một cách tương tự, bit khóa

đọc/ghi chung (bit khóa chế độ 3) không điều khiển việc đọc hoặc viết bằng lệnh LPM/SPM. Nếu nó đang được khắc phục.

**Table 108. Boot Lock Bit0 Protection Modes (Application Section)<sup>(1)</sup>**

BLB0 mode	BLB02	BLB01	Protection
1	1	1	No restrictions for SPM or LPM accessing the Application section.
2	1	0	SPM is not allowed to write to the Application section.
3	0	0	SPM is not allowed to write to the Application section, and LPM executing from the Boot Loader section is not allowed to read from the Application section. If interrupt vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
4	0	1	LPM executing from the Boot Loader section is not allowed to read from the Application section. If interrupt vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.

Note: 1. "1" means unprogrammed, "0" means programmed

**Table 109. Boot Lock Bit1 Protection Modes (Boot Loader Section)<sup>(1)</sup>**

BLB1 mode	BLB12	BLB1 1	Protection
1	1	1	No restrictions for SPM or LPM accessing the Boot Loader section.
2	1	0	SPM is not allowed to write to the Boot Loader section.
3	0	0	SPM is not allowed to write to the Boot Loader section, and LPM executing from the Application section is not allowed to read from the Boot Loader section. If interrupt vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.
4	0	1	LPM executing from the Application section is not allowed to read from the Boot Loader section. If interrupt vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.

Note: 1. "1" means unprogrammed, "0" means programmed

## Việc truy nhập vào chương trình bộ tải

Việc truy nhập vào khởi động bộ tải xảy ra bằng một lệnh nhảy hoặc gọi từ chương trình ứng dụng. Điều này có thể được khởi tạo bởi trigger như là một lệnh được nhận thông qua USART hoặc giao diện SPI. Như một sự luân phiên, cầu chì khởi động reset có thể được lập trình vì vậy vector reset khởi động lại đang đánh dấu tới bộ khởi động flash địa chỉ khởi động sau khi có vector reset. Trong trường hợp này khởi động bộ tải được bắt đầu sau khi có một tín hiệu reset. Sau khi mã ứng dụng được tải chương trình có thể bắt đầu thực thi mã ứng dụng. Chú ý rằng các cầu chì không thể thay đổi bằng bản thân MCU. Điều này có nghĩa là mỗi lần cầu chì khởi động reset được lập trình, vector reset sẽ luôn được lập tới tín hiệu reset khởi động bộ tải và cầu chì chỉ có thể thay đổi thông qua giao diện lập trình song song hoặc nối tiếp

**Table 110. Boot Reset Fuse<sup>(1)</sup>**

BOOTRST	Reset Address
1	Reset Vector = Application Reset (address \$0000)
0	Reset Vector = Boot Loader Reset (see <a href="#">Table 112 on page 284</a> )

Note: 1. "1" means unprogrammed, "0" means programmed

## Thanh ghi trạng thái và bộ nhớ điều khiển chương trình lưu trữ - SPMCSR

Thanh ghi trạng thái và bộ nhớ điều khiển chương trình lưu trữ - SPMCSR bao gồm các bít điều khiển cần thiết để điều khiển hoạt động khởi động bộ tải

Bit	7	6	5	4	3	2	1	0	SPMCSR
Read/Write	SPMIE	RWWSB	-	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	
Initial Value	0	0	0	0	0	0	0	0	

Bít 7-SPMIE: kích hoạt ngắt SPM

Khi bit SPMIE được viết là 1, và bit I trong thanh ghi trạng thái được đặt là 1. SPM sẵn sàng ngắt sẽ được kích hoạt. Ngắt sẵn sàng SPM sẽ được thực thi chỉ cần bit SPMEN trong thanh ghi SPMCSR bị xóa.

Bít 6 – RWWSB: Read-Whlie-Write Section Busy

Khi một sự lập trình (viết trang hoặc ghi trang) điều khiển tới khu vực RWW được khởi tạo, RWWSB được đặt là bằng phần cứng. Khi bit RWWSB được cài đặt, khu vực RWW có thể không được truy cập. bit RWWSB bị xóa nếu bit RvWWSRE được ghi là 1 sau khi một hoạt động lập trình vừa hoàn thành. Như một sự luân phiên, bit RvWWSB sẽ tự động xóa nếu 1 trang tải chế độ khởi động được tải khởi tạo.

Bít 5- RES: bit giữ trữ đây là các bít giữ trữ và luôn luôn được đọc là 0

Bít 4 – RWWSRWE: kích hoạt việc đọc khu vực đọc trong khi viết

Khi lập trình (viết trang hoặc xóa trang). Để kích hoạt lại khu vực RWW, phần mềm người sử dụng phải đợi cho đến khi phần mềm được hoàn tất (SPMEN bị xóa). Sau đó nếu bít RWWSRE được ghi là một tại cùng một thời điểm với bít SPMEN, lệnh SPM tiếp theo trong vòng 4 chu kỳ xung nhịp kích hoạt lại khu vực RWW. Vùng RWW có thể được kích hoạt lại trong khi bộ nhớ flash bận với một quá trình xóa trang hoặc ghi trang (SPMEN được cài đặt). Nếu bít RWWSRE được viết trong khi bộ nhớ flash đang được tải, quá trình điều khiển tải bộ nhớ flash sẽ bỏ qua và giữ liệu được tải sẽ bị mất.

Bít 3- BLBSET : Cài đặt bít khóa khởi động

Nếu bít này được viết là cùng một thời điểm như SPMEN, lệnh SPM tiếp theo trong vòng 4 chu kỳ xung nhịp cài đặt các bit khởi động khóa, theo giữ liệu trong R. Giữ liệu trong R1 và địa chỉ trong con trỏ ngăn xếp Z bị bỏ qua. Bit BLBSET sẽ tự động bị xóa khi hoàn thành việc cài đặt các bit khóa, hoặc nếu lệnh SBM sẽ thực thi trong vòng 4 xung nhịp

Một lệnh LPM trong vòng 3 chu kỳ xung nhịp sau khi bít BLBSET và FPMEN được cài đặt trong thanh ghi SPMCSR, sẽ đọc các bit khóa hoặc các bít cầu chì (phụ thuộc vào bit Z0 của con trỏ ngăn xếp Z) vào trong thanh ghi đích đến. Xem trang 281 Bít 2 – PGWRT nếu bít này được biết là một tại cùng thời điểm như SPMEN , lệnh SPM tiếp theo trong vòng 4 chu kỳ xung nhịp thực thi việc ghi trang, với dữ liệu được lưu trong bộ đệm tạm thời địa chỉ trang được tạo ra từ các phần cao của con trỏ ngăn xếp Z. dữ liệu trong R1 và R0 bị bỏ qua. Bít PRWRT sẽ tự động xóa khi hoàn thành một việc ghi trang, hoặc nếu không có lệnh SPM được thực thi trong vòng 4 chu kỳ xung nhịp. CPU bị dừng trong suốt toàn bộ quá trình ghi trang nếu khu vực NRWW được đánh địa chỉ.

Bít 1- PGERS : Xóa trang nếu bit này được viết là một tại thời điểmt như SPMEN, lệnh SPM tiếp theo trong vòng 4 chu kỳ xung nhịp sẽ thực thi việc xóa trang. Địa chỉ của trang được tạo ra từ phần cao của con trỏ ngăn xếp Z dữ liệu trong R1 và R0 bị bỏ qua Bít PGERS sẽ tự động xóa khi hoàn thành một việc ghi trang, hoặc nếu không có lệnh SPM được thực thi trong vòng 4 chu kỳ xung nhịp. CPU bị dừng trong suốt toàn bộ quá trình ghi trang nếu khu vực NRWW được đánh địa chỉ.

Bit 0- SPMEN : kích hoạt vùng nhớ chương trình lưu trữ

bit này kích hoạt lệnh SPM cho 4 chu kỳ xung nhịp tiếp theo. Nếu được viết là 1 cung với các bit RWWSRE , BLBSET PGWRT, PGERS, lệnh SPM kế tiếp sẽ có một ý nghĩa đặc biệt, như được miêu tả bên dưới. nếu chỉ có SPMEN được viết, lệnh SPM kế tiếp sẽ được lưu giá trị trong R1:R0 trong bộ đệm đại chỉ trang tạm thời bởi con trỏ ngăn xếp Z. LSB của con trỏ ngăn xếp Z được bỏ qua. Bit SPMEN sẽ tự động xoa khi hoàn thành một lệnh SPM hoặc nếu không có lệnh SPM được thực thi trong vòng 4 chu kỳ xung nhịp. trong suốt quá trình xóa trang hoặc ghi trang, bít SPMEN luôn ở mức cao cho đến khi hoạt động được hoàn tất.

Việc viết bất cứ sự kết nối khác nào hơn “ 10001”, 01001, 00101, 00011, 00001 trong 5 bit thấp hơn thì không có hiệu lực

### **Việc đặt địa chỉ Flash trong suốt quá trình tự lập trình .**

Con trỏ ngăn xếp Z với RAMPZ được sử dụng để đánh địa chỉ các lệnh SPM . Về chi tiết cách sử dụng RAMPZ , xem phần thanh ghi lựa chọn RAM page Z ở trang 14 .

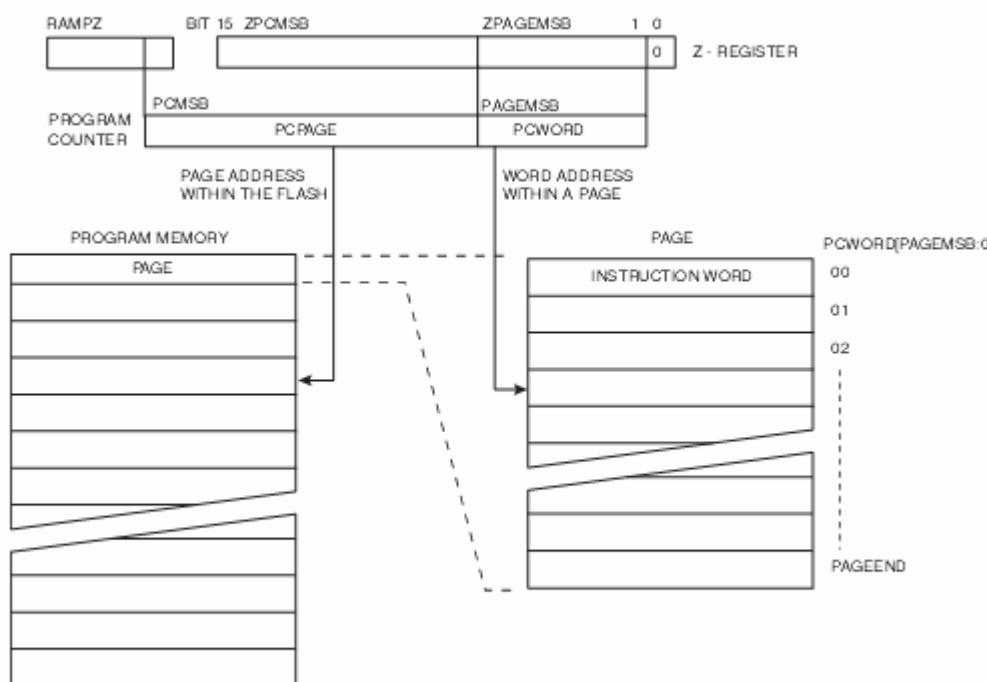
Bit	15	14	13	12	11	10	9	8
ZH (R31)	Z15	Z14	Z13	Z12	Z11	Z10	Z9	Z8
ZL (R30)	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0

Do bộ nhớ Flash được tổ chức trong dạng trang (xem bảng 123 trang 291)bộ đếm chương trình có thể được xử lý như là có 2 vùng nhớ khác nhau .Một khu vực thì bao gồm các bit có trọng số nhỏ nhất ,được đánh địa chỉ các từ trong vòng một trang, trong khi các bít có trọng số lớn nhất được đánh địa chỉ các trang .Điều này được chỉ ra trong hình 134 .Chú ý rằng quá trình xóa trang hoặc ghi trang thì được đánh địa chỉ một cách độc lập .Vì vậy nó là một thành phần chính quan trọng mà phần mềm khởi động bộ tải đánh địa chỉ các trang giống nhau trong cả 2 quá trình ghi trang và xóa trang

Mỗi lần một quá trình lập trình được khởi tạo ,địa chỉ được chốt và con trỏ z /RAMPZ có thể được sử dụng cho các chế độ điều khiển khác .

Chỉ có hoạt động SPM cái mà không sử dụng con trỏ s/RAMPZ để lưu trữ địa chỉ .Do các địa chỉ của lệnh này đánh địa chỉ các byte flat bằng byte ,cũng sử dụng các bít LSB( bit z0 )của con trỏ z

**Figure 134. Addressing the Flash During SPM<sup>(1)</sup>**



Notes: 1. The different variables used in Figure 134 are listed in Table 114 on page 285.  
2. PCPAGE and PCWORD are listed in Table 124 on page 292.

## Tự lập trình bộ nhớ flash

Bộ nhớ chương trình được cập nhật trong một trang bằng page fashion.Trước khi quá trình lập trình một trang với dữ liệu được lưu trong bộ đệm trang tạm thời , trang phải được xóa . Bộ đệm trang tạm thời được điền đầy một từ tại một thời điểm sử dụng SPM và bộ đệm có thể được điền đầy trước khi có lệnh xóa trang hoặc giữa 1 quá trình xóa trang và lưu trang :

Phương án 1 , điền đầy bộ đệm trước khi 1 trang bị xóa

- Điền đầy bộ đệm trang
- Tiến hành một việc xóa trang
- Tiến hành một việc viết trang

Phương án 2 , điền đầy bộ đệm sau khi xóa trang

- tiến hành 1 sự xóa trang
- điền đầy bộ đệm trang
- Tiến hành một quá trình viết trang

Nếu chỉ 1 phần của trang cần thiết phải thay đổi , phần còn lại của trang phải được lưu lại (ví dụ như trong bộ đệm trang tạm thời ) trước khi xóa , và sau đó được viết lại . Khi sử dụng phương án 1 , bộ tải khởi động cung cấp một đặc điểm Read – Modify-Write có ích cái mà cho phép phần mềm người sử dụng đọc trước tiên trang , thực hiện các thay đổi cần thiết , và sau đó ghi trở lại các dữ liệu đã sửa đổi . Nếu phương án 2 được sử dụng , nó không thể đọc các dữ liệu cũ trong khi đang tải các do đó trang thì sẵn sàng được xóa . Bộ đệm trang tạm thời có thể được truy cập trong một chuỗi ngẫu nhiên . Điều cốt yếu là địa chỉ trang được sử dụng trong cả 2 quá trình đọc trang và xóa trang thì đang đánh địa chỉ các trang giống nhau .Xem trang 282 .

## Tiến hành việc xoá trang bằng SPM

Để thực thi việc xoá trang cài đặt địa chỉ trong con trỏ z và dữ liệu trong R1 :R0 ,viết “ 00000001”lên SPMCSR và thực thi SPM trong vòng 4 chu kỳ xung nhịp sau khi viết SPMCSR . Thành phần của PCWORD trong thanh ghi z được sử dụng để đánh địa chỉ dữ liệu trong bộ đệm tạm thời .Bộ đệm tạm thời sẽ tự động xóa sau khi một quá trình viết trang hoặc sự viết bit RWWSRE Trong thanh ghi SPMCSR .Nó cũng bị xóa sau khi reset hệ thống .Chú ý rằng không thể viết nhiều hơn một lần lên mỗi địa chỉ thiếu sự xóa bộ đệm tạm thời .chú ý nếu EEPROM được ghi ở giữa của quá trình tải trang SPM ,tất cả giữ liệu được tải sẽ bị mất

## Tiến hành viết một trang

Để thực thi việc viết trang, cài đặt địa chỉ trong con trỏ z và RAMPZ ,viết “X0000101” lên SPMCSR và thực thi SPM trong vòng 4 chu kỳ xung nhịp sau khi viết SPMCSR . Dữ liệu trong R1 và R0 được bỏ qua .địa chỉ trang phải được viết tới PCPAGE .Các bít khác trong con trỏ z phải được viết là 0 trong suốt quá trình này

- Viết trang tới khu vực RWW :khu vực NRWW có thể được đặt trong suốt quá trình viết trang
- Viết trang lên khu vực NRWW : CPU bị dừng trong suốt quá trình điều khiển

## Sử dụng ngắt SPM

Nếu ngắt SPM được kích hoạt ,ngắt SPM sẽ sinh ra một ngắt không đổi khi bít SPM trong thanh ghi SPMCSR bị xóa .Điều này có nghĩa là ngắt có thể sử dụng sự hỏi vòng thanh ghi SPMCSR trong phần mềm .Khi sử dụng ngắt SPM ,các vecto ngắt nên được di chuyển nên khu vực BMS để tránh việc một ngắt đang truy nhập khu vực RWW khi nó bị khóa cho việc đọc .Cách để di chuyển các ngắt thì được miêu tả trong trang 60

## Sự xét đến trong khi cập nhật BLS

Một sự cần trọng đặc biệt phải được tuân theo nếu người sử dụng cho phép khu vực khởi động bộ tải dữ liệu được cập nhật bằng việc di chuyển bít khóa số 11 không được lập trình một tai nạn trong khi viết tới bản thân bộ khởi động tải có thể làm hư hỏng toàn bộ bộ tải khởi động ,và sự cập nhật phần mềm khác có thể bị dán đoạn .Nếu không thực sự cần thiết để thay đổi phần mềm khởi động bộ tải ,nó được đề nghị để lập trình bít khóa 11 để bảo vệ phần mềm khởi động bộ tải từ bất cứ thay đổi phần mềm bên trong nào

## Ngăn cản việc đọc khu vực RWW trong suốt quá trình tự lập trình

Trong suốt quá trình tự lập trình ( xóa trang hoặc viết trang),khu vực RWW

Thì luôn bị cô lập trong việc đọc .bản thân phần mềm người sử dụng phải ngăn cản cái mà khu vực này được đánh địa chỉ trong suốt quá trình tự lập trình bít RWWSB trong thanh ghi SPMCSR sẽ được cài đặt chỉ cần khu vực RWW đang bận .Trong suốt quá trình tự lập trình bằng vecto ngắt nên được di chuyển tới BLS như được miêu tả ở trang 60 ,hoặc các ngắt phải bị vô hiệu hóa .Trước khi đánh địa chỉ RWW sau khi sự lập trình hoàn tất ,phần mềm người dùng phải xóa RWWSB bằng việc viết RWWSRE. Xem trang 282 như một ví dụ

## Cài đặt các bít khóa khởi động bộ tải bằng SPM

Để cài đặt các bít khóa khởi động tại ,viết dữ liệu thích đáng tới R0,viết “X0001001” lên thanh ghi SBMCSR và thực thi SPM trong vòng 4 chu kỳ xung nhiệt sau khi viết SBMCSR .chỉ có thể truy cập các bít khóa là các bít khóa khởi động cái mà có thể ngăn cản khu vực ứng dụng và khởi động bộ tải từ bất cứ sự cập nhật phần mềm nào bởi MCU

Bit	7	6	5	4	3	2	1	0
R0	1	1	BLB12	BLB11	BLB02	BLB01	1	1

Xem bảng 108 và 109 về cách cài đặt khác nhau của các bít khởi động bộ tải ảnh hưởng đến sự truy cập bộ nhớ flash .Nếu bít 5 ... 2 trong R0 bị xóa (0)bít khóa khởi động tương ứng sẽ được lập trình nếu một lệnh SPM được thực thi trong vòng 4 chu kỳ xung nhịp sau khi BLBSET và SPMEN được cài đặt trong SPMCSR .Con trỏ z thì không cần thận trong suốt quá trình điều khiển này nhưng để tương thích trong tương lai nó được đề nghị để tải con trỏ z với \$0001 (giống như được sử dụng cho việc đọc các bít xóa ).để tương thích trong tương lai nó cũng được khuyến cáo để cài đặt các bít

,7,6,1,và 0 trong R0 để giá trị 1 khi viết các bít khóa .Khi lập trình các bít khóa toàn bộ bộ nhớ flash có thể được đọc trong suốt quá trình .

## Viết EEPROM ngăn cản quá trình viết nén SPMCSR

Chú ý rằng một quá trình viết EEPROM sẽ cách ly phần mềm lập trình flash việc đọc các bit cầu chì và các bit khóa từ phần mềm sẽ ngăn cản trong suốt quá trình viết EEPROM điều này được khuyến cáo rằng người sử dụng kiểm tra bit trạng thái (EEWE) trong thanh ghi EECR và phân tích rằng bit bị xóa trước khi viết lên thanh ghi SPMCSR .

### Đọc cầu chì và các bit khóa từ phần mềm

Có thể đọc cả 2 bit cầu chì và các bit khóa từ phần mềm. Để đọc các bit khóa tải con trỏ Z với \$ 0001 và cài đặt các bit BLBSET và SPMEN trong thanh ghi SPMCSR. Khi một lệnh LPM được thực thi trong vòng 3 chu kỳ xung nhịp CPU sau khi các bit BLBSET và SPMEN được cài đặt trong SPMCSR, giá trị của các bit khóa sẽ được tải và trong thanh ghi đích. Các bit BLBSET và SPMEN sẽ tự động bị xóa trên một sự hoàn thành của việc đọc các bit khóa hoặc nếu không có lệnh MPL nào được thực thi trong vòng 3 chu kỳ xung nhịp CPU hoặc không có lệnh SPM nào được thực thi trong 4 chu kỳ xung nhịp CPU. Khi BLBSET và SPMEN bị xóa , LPM sẽ làm việc như được miêu tả trong hướng dẫn cài đặt lệnh.

Bit	7	6	5	4	3	2	1	0
Rd	FLB7	FLB6	FLB5	FLB4	FLB3	FLB2	FLB1	FLB0

Thuật toán cho việc đọc các bit cầu chì thấp, tải con trỏ Z với \$ 0000 và cài đặt các bit BLBSET và SPMEN trong thanh ghi SPMCSR. Khi một lệnh LPM được thực thi trong vòng 3 chu kỳ xung nhịp sau khi các bit BLBSET và SPMEN được cài đặt trong SPMCSR, giá trị của các bit cầu chì thấp ( FLB) sẽ được tải vào trong thanh ghi đích đến như được chỉ ra dưới đây. Tham khảo bảng 119 trang 288 để có sự miêu tả chi tiết và bản đồ của các bit cầu chì thấp

Bit	7	6	5	4	3	2	1	0
Rd	FLB7	FLB6	FLB5	FLB4	FLB3	FLB2	FLB1	FLB0

Một cách tương tự khi đọc các bit cầu chì cao tải \$ 0003 trong con trỏ Z. khi một lệnh LPM được thực thi trong vòng 3 chu kỳ xung nhịp sau khi các bit BLBSET và SPMEN được cài đặt trong SPMCSR, giá trị của các bit cầu chì cao ( FHP) sẽ được tải vào trong thanh ghi đích đến như được chỉ ra bên dưới. Tham khảo bảng 118 trang 288 để biết sự miêu tả chi tiết và bản đồ các bit cầu chì cao

Bit	7	6	5	4	3	2	1	0
Rd	FHB7	FHB6	FHB5	FHB4	FHB3	FHB2	FHB1	FHB0

Khi viết các bít cầu chì mở rộng, tải \$ 0002 vào trong con trỏ Z .. khi một lệnh LPM được thực thi trong vòng 3 chu kỳ xung nhịp sau khi các bit BLBSET và SPMEN được cài đặt trong SPMCSR, giá trị của các bit cầu chì mở rộng ( EFB) sẽ được tải vào

trong thanh ghi đích đến như được chỉ ra bên dưới. Tham khảo bảng 117 trang 287 để biết sự miêu tả chi tiết và bản đồ các bit cầu chì cao.

Bit Rd	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	EFB1	EFB0

Cầu chì và các bit khóa được lập trình sẽ được đọc là 0. Cầu chì và các bit khóa không được lập trình sẽ được đọc là 1.

### Sự ngăn cản việc hỏng bộ nhớ flash

Trong suốt chu kỳ của VCC thấp chương trình bộ nhớ flash có thể bị hư hỏng bởi vì điện áp nguồn cấp thấp quá thấp cho CPU và bộ nhớ flash hoạt động bình thường các sự ban hành này thì giống như cho cấp bo mạch hệ thống sử dụng bộ nhớ flash, và môi trường thiết kế giống nhau nên được áp dụng.

Một sự hỏng hóc bộ nhớ flash có thể được gây ra bởi hai trường hợp khi điện áp quá thấp. Đầu tiên, 1 chuỗi viết thông thường tới bộ nhớ flash yêu cầu 1 điện áp cực tiểu để hoạt động bình thường. Thứ hai, bản thân CPU có thể thực thi các lệnh không chính xác, nếu điện áp cung cấp cho việc thực thi các lệnh quá thấp.

Sự hư hỏng bộ nhớ flash có thể dễ dàng tránh được bằng cách đề nghị thiết kế dưới đây ( chỉ một là đủ).

1. nếu không cần thiết có một bộ cập nhật khởi động bộ tải trong hệ thống, các bit khóa khởi động bộ tải ngăn cản bắt cứ sự cập nhật phần mềm khởi động bộ tải
2. giữ cho hoạt động reset AVR trong suốt các chu kỳ của sự không đủ điện áp nguồn cung cấp. Điều này có thể được thực hiện bằng cách kích hoạt bộ dò thiếu điện áp bên trong (BOD) nếu như điện áp điều khiển tương ứng với mức dò. Nếu không, một mạch bảo vệ reset VCC thấp bên ngoài có thể được sử dụng. nếu một tín hiệu reset xuất hiện trong khi một quá trình viết đang tiến hành, quá trình viết sẽ được hoàn thành cung cấp điện áp nguồn là đủ
3. giữ cho lõi AVR ở chế độ ngủ Power-on trong suốt chu kỳ của VCC thấp. Điều này sẽ ngăn cản CPU khỏi việc cố gắng để mã hóa lại và thực thi các lệnh, việc bảo vệ hiệu quả thanh ghi SPMCER và do đó bảo vệ bộ nhớ flash khỏi các quá trình ghi không mong muốn.

thời gian lập trình cho bộ nhớ flash khi sử dụng SPM.

Bộ tạo giao động RC hiệu chỉnh được sử dụng để đo thời gian truy nhập flash. Bảng 111 chỉ ra thời gian lập trình thông thường cho sự truy nhập flash từ CPU.

Table 111. SPM Programming Time.

Symbol	Min Programming Time	Max Programming Time
Flash write (page erase, page write, and write lock bits by SPM)	3.7 ms	4.5 ms

### Ví dụ mẫu đơn giản đoạn code Assembly cho bộ khởi động bộ tải

```

;-the routine writes one page of data from RAM to Flash
; the first data location in RAM is pointed to by the Y pointer
; the first data location in Flash is pointed to by the Z-pointer
;-error handling is not included
;-the routine must be placed inside the boot space
; (at least the Do_spm sub routine). Only code inside NRWW section can
; be read during self-programming (page erase and page write).
;-registers used: r0, r1, temp1 (r16), temp2 (r17), looplo (r24),
; loophi (r25), spmcsvval (r20)
; storing and restoring of registers is not included in the routine
; register usage can be optimized at the expense of code size
;-It is assumed that either the interrupt table is moved to the Boot
; loader section or that the interrupts are disabled.
.equ PAGESIZEB = PAGESIZE*2      ;PAGESIZEB is page size in BYTES, not words
.org SMALLBOOTSTART

Write_page:
    ; page erase
    ldi spmcsvval, (1<<PGERS) | (1<<SPMEN)
    call Do_spm

    ; re-enable the RW section
    ldi spmcsvval, (1<<RWWSRE) | (1<<SPMEN)
    call Do_spm

    ; transfer data from RAM to Flash page buffer
    ldi looplo, low(PAGESIZEB);init loop variable
    ldi loophi, high(PAGESIZEB);not required for PAGESIZEB<=256
Wrloop:
    ld r0, Y+
    ld r1, Y+
    ldi spmcsvval, (1<<SPMEN)
    call Do_spm
    adiw ZH:ZL, 2
    sbiw loophi:looplo, 2      ;use subi for PAGESIZEB<=256
    brne Wrloop

    ; execute page write
    subi ZL, low(PAGESIZEB)    ;restore pointer
    sbci ZH, high(PAGESIZEB)   ;not required for PAGESIZEB<=256
    ldi spmcsvval, (1<<PGWRT) | (1<<SPMEN)
    call Do_spm

    ; re-enable the RW section
    ldi spmcsvval, (1<<RWWSRE) | (1<<SPMEN)
    call Do_spm

    ; read back and check, optional
    ldi looplo, low(PAGESIZEB);init loop variable
    ldi loophi, high(PAGESIZEB);not required for PAGESIZEB<=256
    subi YL, low(PAGESIZEB)    ;restore pointer
    sbci YH, high(PAGESIZEB)
Rdloop:
    lpm r0, Z+
    ld r1, Y+
    cpse r0, r1
    jmp Error
    sbiw loophi:looplo, 1      ;use subi for PAGESIZEB<=256
    brne Rdloop

```

```

; return to RWW section
; verify that RWW section is safe to read
Return:
lds temp1, SPMCSR
sbrs temp1, RWWSB           ; If RWWSB is set, the RWW section is not ready
yet
ret
; re-enable the RWW section
ldi spmcsrvval, (1<<RWWSRE) | (1<<SPMEN)
call Do_spm
rjmp Return

Do_spm:
; check for previous SPM complete
Wait_spm:
lds temp1, SPMCSR
sbrc temp1, SPMEN
rjmp Wait_spm
; input: spmcsrvval determines SPM action
; disable interrupts if enabled, store status
in temp2, SREG
cli
; check that no EEPROM write access is present

Wait_ee:
sbic EECR, EEWE
rjmp Wait_ee
; SPM timed sequence
sts SPMCSR, spmcsrvval
spm
; restore SREG (to enable interrupts if originally enabled)
out SREG, temp2
ret

```

## Các tham số khởi động bộ tải Atmega 128

Trong bảng 112 đến bảng 114 , các tham số được sử dụng trong phần miêu tả của sự tự lập trình được đưa ra

Table 112. Boot Size Configuration

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application section	Boot Reset Address (start Boot Loader Section)
1	1	512 words	4	\$0000 - \$FDFF	\$FE00 - \$FFFF	\$FDFF	\$FE00
1	0	1024 words	8	\$0000 - \$FBFF	\$FC00 - \$FFFF	\$FBFF	\$FC00
0	1	2048 words	16	\$0000 - \$F7FF	\$F800 - \$FFFF	\$F7FF	\$F800
0	0	4096 words	32	\$0000 - \$EFFF	\$F000 - \$FFFF	\$EFFF	\$F000

Note: The different BOOTSZ fuse configurations are shown in Figure 133

**Table 113.** Read-While-Write Limit<sup>(1)</sup>

Section	Pages	Address
Read-While-Write section (RWW)	480	\$0000 - \$EFFF
No Read-While-Write section (NRWW)	32	\$F000 - \$FFFF

Note: 1. For details about these two sections, see "No Read-While-Write Section – NRWW" on page 274 and "Read-While-Write Section – RWW" on page 274

**Table 114.** Explanation of Different Variables Used in [Figure 134](#) and the Mapping to the Z-Pointer<sup>(3)</sup>

Variable		Corresponding Z-value	Description <sup>(2)</sup>
PCMSB	15		Most significant bit in the program counter. (The program counter is 16 bits PC[15:0])
PAGEMS8	6		Most significant bit which is used to address the words within one page (128 words in a page requires 7 bits PC [6:0]).
ZPCMSB		Z16 <sup>(1)</sup>	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPCMSB equals PCMSB + 1.
ZPAGEMS8		Z7	Bit in Z-register that is mapped to PAGEMS8. Because Z0 is not used, the ZPAGEMS8 equals PAGEMS8 + 1.
PCPAGE	PC[15:7]	Z16 <sup>(1)</sup> :Z8	Program counter page address: Page select, for page erase and page write
PCWORD	PC[6:0]	Z7:Z1	Program counter word address: Word select, for filling temporary buffer (must be zero during page write operation)

Notes: 1. The Z-register is only 16 bits wide. Bit 16 is located in the RAMPZ register in the I/O map.  
 2. Z0: should be zero for all SPM commands, byte select for the (E)LPM instruction.  
 3. See "[Addressing the Flash During Self-Programming](#)" on page 278 for details about the use of Z-pointer during self-programming.

## XXIV . Lập trình bộ nhớ - Memory Programming

### Các bit khóa dữ liệu bộ nhớ và chương trình

Atmega 128 cung cấp 6 bit khóa cái mà có thể cái mà không thể lập trình (1) hoặc có thể được lập trình (0) để đạt được các đặc tính được thêm vào như được liệt kê trong bảng 116. Các bit khóa chỉ có thể bị xóa lên 1 với lệnh xóa chip – chip eraser command

**Table 115. Lock Bit Byte**

Lock Bit Byte	Bit No.	Description	Default Value
	7	–	1 (unprogrammed)
	6	–	1 (unprogrammed)
BLB12	5	Boot lock bit	1 (unprogrammed)
BLB11	4	Boot lock bit	1 (unprogrammed)
BLB02	3	Boot lock bit	1 (unprogrammed)
BLB01	2	Boot lock bit	1 (unprogrammed)
LB2	1	Lock bit	1 (unprogrammed)
LB1	0	Lock bit	1 (unprogrammed)

Note: "1" means unprogrammed, "0" means programmed

**Table 116. Lock Bit Protection Modes**

Memory Lock Bits			Protection Type
LB mode	LB2	LB1	
1	1	1	No memory lock features enabled.
2	1	0	Further programming of the Flash and EEPROM is disabled in Parallel and SPI/JTAG Serial Programming mode. The Fuse bits are locked in both Serial and Parallel Programming mode. <sup>(1)</sup>
3	0	0	Further programming and verification of the Flash and EEPROM is disabled in Parallel and SPI/JTAG Serial Programming mode. The Fuse bits are locked in both Serial and Parallel Programming mode. <sup>(1)</sup>
BLB0 mode	BLB02	BLB01	
1	1	1	No restrictions for SPM or (E)LPM accessing the Application section.
2	1	0	SPM is not allowed to write to the Application section.
3	0	0	SPM is not allowed to write to the Application section, and (E)LPM executing from the Boot Loader section is not allowed to read from the Application section. If interrupt vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
4	0	1	(E)LPM executing from the Boot Loader section is not allowed to read from the Application section. If interrupt vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
BLB1 mode	BLB12	BLB11	

**Table 116.** Lock Bit Protection Modes (Continued)

Memory Lock Bits			Protection Type
1	1	1	No restrictions for SPM or (E)LPM accessing the Boot Loader section.
2	1	0	SPM is not allowed to write to the Boot Loader section.
3	0	0	SPM is not allowed to write to the Boot Loader section, and (E)LPM executing from the Application section is not allowed to read from the Boot Loader section. If interrupt vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.
4	0	1	(E)LPM executing from the Application section is not allowed to read from the Boot Loader section. If interrupt vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.

Notes: 1. Program the fuse bits before programming the Lock bits.

2. "1" means unprogrammed, "0" means programmed

## Các bit cầu chì - Fuse bits

Atmega 128 có 3 byte cầu chì . Bảng 117 – 119 miêu tả ngắn gọn chức năng của tất cả các bit cầu chì và cách chúng được vẽ bản đồ bên trong bytes cầu chì . Chú ý rằng các cầu chì được đọc như là mức logic 0 , nếu chúng được lập trình .

**Table 117.** Extended Fuse Byte

Extended Fuse Byte	Bit No.	Description	Default Value
-	7	-	1
-	6	-	1
-	5	-	1
-	4	-	1
-	3	-	1
-	2	-	1
M103C <sup>(1)</sup>	1	ATmega103 compatibility mode	0 (programmed)
WDTON <sup>(2)</sup>	0	Watchdog Timer always on	1 (unprogrammed)

Notes: 1. See "ATmega103 and ATmega128 Compatibility" on page 4 for details.

2. See "Watchdog Timer Control Register – WDTCR" on page 56 for details.

**Table 118.** Fuse High Byte

Fuse High Byte	Bit No.	Description	Default Value
OCDEN <sup>(4)</sup>	7	Enable OCD	1 (unprogrammed, OCD disabled)
JTAGEN <sup>(5)</sup>	6	Enable JTAG	0 (programmed, JTAG enabled)
SPIEN <sup>(1)</sup>	5	Enable Serial Program and Data Downloading	0 (programmed, SPI prog. enabled)
CKOPT <sup>(2)</sup>	4	Oscillator options	1 (unprogrammed)
EESAVE	3	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed, EEPROM not preserved)
BOOTSZ1	2	Select Boot Size (see <a href="#">Table 112</a> for details)	0 (programmed) <sup>(3)</sup>
BOOTSZ0	1	Select Boot Size (see <a href="#">Table 112</a> for details)	0 (programmed) <sup>(3)</sup>
BOOTRST	0	Select Reset Vector	1 (unprogrammed)

- Notes:
1. The SPIEN fuse is not accessible in SPI Serial Programming mode.
  2. The CKOPT fuse functionality depends on the setting of the CKSEL bits. See "Clock Sources" on page 37 for details.
  3. The default value of BOOTSZ1..0 results in maximum Boot Size. See [Table 112 on page 284](#).
  4. Never ship a product with the OCDEN Fuse programmed regardless of the setting of lock bits and the JTAGEN Fuse. A programmed OCDEN Fuse enables some parts of the clock system to be running in all sleep modes. This may increase the power consumption.
  5. If the JTAG interface is left unconnected, the JTAGEN fuse should if possible be disabled. This to avoid static current at the TDO pin in the JTAG interface.

**Table 119.** Fuse Low Byte

Fuse Low Byte	Bit No.	Description	Default Value
BODLEVEL	7	Brown out detector trigger level	1 (unprogrammed)
BODEN	6	Brown out detector enable	1 (unprogrammed, BOD disabled)
SUT1	5	Select start-up time	1 (unprogrammed) <sup>(1)</sup>
SUT0	4	Select start-up time	0 (programmed) <sup>(1)</sup>
CKSEL3	3	Select Clock source	0 (programmed) <sup>(2)</sup>
CKSEL2	2	Select Clock source	0 (programmed) <sup>(2)</sup>
CKSEL1	1	Select Clock source	0 (programmed) <sup>(2)</sup>
CKSEL0	0	Select Clock source	1 (unprogrammed) <sup>(2)</sup>

- Notes:
1. The default value of SUT1..0 results in maximum start-up time. See [Table 14 on page 42](#) for details.
  2. The default setting of CKSEL3..0 results in Internal RC Oscillator @ 1 MHz. See [Table 6 on page 37](#) for details.

Trạng thái của các bit cầu chì thì không bị ảnh hưởng bởi lệnh xóa chip . Chú ý rằng các bit cầu chì bị khóa nếu bit khóa 1 (LB1) được lập trình . Lập trình các bit cầu chì trước khi lập trình các bit khóa .

## Quá trình chốt các bit cầu chì

Các giá trị của cầu chì được chốt khi thiết bị truy nhập vào chế độ lập trình và các thay đổi giá trị của các cầu chì sẽ không có hiệu lực cho đến khi phần đó rời khỏi chế độ lập trình . Điều này không được áp dụng lên cầu chì EESAVE cái mà sẽ làm ảnh hưởng mỗi lần nó được lập trình . Các cầu chì thì cũng bị chốt power- up trong chế độ bình thường

## Các byte kí hiệu

Tất cả các vi điều khiển của Atmel có 3 byte mã kí hiệu cái mà dùng để nhận ra thiết bị

Mã này có thể được đọc trong 2 chế độ nối tiếp và song song , cũng đọc khi mà thiết bị bị khóa . 3 bytes này lưu trú trong 1 không gian địa chỉ riêng biệt

Về cho Atmega 128 thì các byte kí hiệu là :

1. \$000:\$1E ( chứng tỏ rằng được thiết kế bởi Atmel )
2. \$001:\$97 ( chứng tỏ rằng có 128KB bộ nhớ Flash 0
3. \$002:\$02( hiển thị rằng thiết bị là Atmega 128 khi \$001 là \$97)

## Byte hiệu chỉnh – Calibration Byte

Atmega 128 lưu 4 giá trị hiệu chỉnh khác nhau cho bộ tạo xung nhịp RC bên trong . Các byte này lưu trú trong các byte kí hiệu cao của các địa chỉ 0x000 , 0x0001, 0x0002 , và 0x0003 cho các thứ tự định sẵn 1 , 2 , 4 , 8 MHz . Trong suốt quá trình Reset , giá trị 1 MHz thì tự động được tải vào trong thanh ghi OSCCAL . Nếu các tần số khác được sử dụng , giá trị hiệu chỉnh phải được tải một cách thông thường, xem phần thanh ghi hiệu chỉnh bộ tạo dao động – OSCCAL trên trang 42 để biết thêm chi tiết .

## Các tham số của quá trình lập trình song song , các chân đánh dấu , và các lệnh

Phần này miêu tả cách để lập trình song song và kiểm tra lại bộ nhớ Flash chương trình , bộ nhớ dữ liệu EEPROM , các bit khóa bộ nhớ , và các bit cầu chì trong Atmega 128 . Các xung được giả định là ở dưới 250ns trừ phi có chú ý khác .

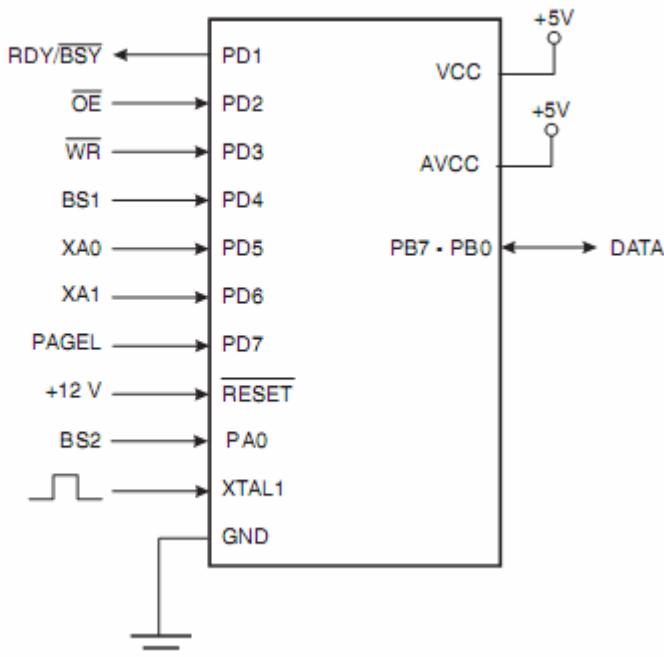
## Tên các tín hiệu

Trong phần này , vài chân của Atmega 128 được tham chiếu bởi các tín hiệu được miêu tả các chức năng của chúng trong suốt quá trình song song , xem hình 135 và bảng 120. các chân thì không được miêu tả trong bảng tiếp theo được tham chiếu bởi tên các chân

Các chân XA1/XA0 xác định hành động được thực thi khi mà chân XTAL1 được đưa ra 1 xung dương . Sự mã hóa bit được chỉ ra trong bảng 122

Khi phát xung WR và OE , lệnh được tải xác định hành động được thực thi . Các lệnh khác được chỉ ra trong 123 .

**Figure 135. Parallel Programming**



**Table 120. Pin Name Mapping**

Signal Name in Programming Mode	Pin Name	I/O	Function
RDY/BSY	PD1	O	0: Device is busy programming, 1: Device is ready for new command
OE	PD2	I	Output Enable (Active low)
WR	PD3	I	Write Pulse (Active low)
BS1	PD4	I	Byte Select 1 ("0" selects low byte, "1" selects high byte)
XA0	PD5	I	XTAL Action Bit 0

**Table 120.** Pin Name Mapping (Continued)

Signal Name in Programming Mode	Pin Name	I/O	Function
XA1	PD6	I	XTAL Action Bit 1
PAGEL	PD7	I	Program Memory and EEPROM data Page Load
BS2	PA0	I	Byte Select 2 ("0" selects low byte, "1" selects 2'nd high byte)
DATA	PB7-0	I/O	Bi-directional Data bus (Output when OE is low)

**Table 121.** Pin Values Used to Enter Programming Mode

Pin	Symbol	Value
PAGEL	Prog_enable[3]	0
XA1	Prog_enable[2]	0
XA0	Prog_enable[1]	0
BS1	Prog_enable[0]	0

**Table 122.** XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load Flash or EEPROM Address (High or low address byte determined by BS1)
0	1	Load Data (High or Low data byte for Flash determined by BS1)
1	0	Load Command
1	1	No Action, Idle

**Table 123.** Command Byte Bit Coding

Command Byte	Command Executed
1000 0000	Chip Erase
0100 0000	Write Fuse bits
0010 0000	Write Lock bits
0001 0000	Write Flash
0001 0001	Write EEPROM
0000 1000	Read Signature Bytes and Calibration byte
0000 0100	Read Fuse and Lock bits
0000 0010	Read Flash
0000 0011	Read EEPROM

**Table 124.** No. of Words in a Page and no. of Pages in the Flash

Flash Size	Page Size	PCWORD	No. of Pages	PCPAGE	PCMSB
64K words (128K bytes)	128 words	PC[6:0]	512	PC[15:7]	15

**Table 125.** No. of Words in a Page and no. of Pages in the EEPROM

EEPROM Size	Page Size	PCWORD	No. of Pages	PCPAGE	EEAMSB
4K bytes	8 bytes	EEA[2:0]	512	EEA[11:3]	8

## XXIV . Lập trình song song

### Truy nhập vào chế độ lập trình song song

Thuật toán tiếp theo,đặt thiết bị vào trong chế độ lập trình song song:

- áp dụng 4,5-5,5V giữa chân V<sub>CC</sub> và chân GND,và đợi ở dưới mức 100micro giây
- cài đặt RESET là 0 để di chuyển XTAL1 dưới 6 lần
- cài đặt prog\_kích hoạt các chân được liệt kê trong bảng 121 trang 291 lên giá trị 0000 và đợi trong khoảng dưới 100 nano giây
- áp dụng điện áp 11 đến 12,5V lên chân RESET.bắt cứ hành động prog nào kích hoạt các chân trong vòng 100nano giây sau khi điện áp +12v được đặt lên chân RESET,sẽ gây ra sự sai hỏng của thiết bị khi truy nhập vào chế độ lập trình.

Chú ý rằng,bộ tạo xung nhịp thạch anh bên ngoài hoặc bộ tạo dao động RC bên ngoài được lựa chọn,nếu nó không thể đặt lên xung chuẩn XTAL1.trong trường hợp như thế,thuật toán sau đây phải được tuân theo:

- đặt prog\_kích hoạt các chân ở bảng 291 lên 0000.
- đặt điện áp 4,5-5,5V giữa chân V<sub>CC</sub> và chân GND đồng thời như là điện áp 11 đến 12,5V lên chân RESET.
- đợi 100nano giây
- lập trình lại các cầu chì để đảm bảo rằng các xung nhịp bên ngoài được lựa chọn như là các nguồn xung nhịp(CKSEL3:0=0b0000) nếu các bit khóa được lập trình,một lệnh xóa chip phải được thực thi trước khi thay đổi các bit cầu chì
- khi thoát ra khỏi chế độ lập trình bằng việc tắt nguồn của thiết bị hoặc mang chân RESET lên giá trị 0b0
- truy nhập vào chế độ lập trình với các thuật toán chính thức như là được miêu tả bên dưới.

## sự xét đến của việc lập trình

lệnh tải và các địa chỉ được lưu lại trong thiết bị này trong suốt quá trình lập trình.để tiện cho việc lập trình,các bước bên dưới đây nên được xét đến

- các lệnh chỉ cần được tải mỗi lần khi viết hoặc đọc nhiều vùng nhớ khác nhau.
- Việc giữ quá trình viết các giá trị dữ liệu \$FF,cái mà bao gồm các giá trị được lưu trú trong EEPROM( trừ phi cầu chì EESAVE được lập trình) và bộ nhớ flash sau khi có lệnh xóa chip.
- Các byte có địa chỉ cao chỉ cần được load trước khi lập trình hoặc đọc một từ mới có 256 byte trong bộ nhớ flash hoặc EEPROM.điều này cũng được xét đến để đọc các byte kí hiệu.

## Xóa chip

Lệnh xóa chip sẽ xóa bộ nhớ flash và bộ nhớ EEPROM và các bit khóa.các bit khóa thì không được reset cho đến khi bộ nhớ chương trình được xoá hoàn toàn.các bit cầu chì thì không bị thay đổi.một lệnh xóa chip phải được tiến hành trước khi bộ nhớ flash hoặc EEPROM được lập trình lại.

Chú ý:bộ nhớ EEPROM được phục vụ trước trong suốt quá trình xóa chip nếu như cầu chì EESAVE được lập trình.

Quá trình tải lệnh xóa chip

1. đặt XA1,XA0 lên 10.điều này kích hoạt việc tải lệnh
2. đặt BS1 là 0
3. đặt DATA lên 1000 0000.đây là lệnh để xóa chip
4. đưa XTAL1 là một xung dương.điều này để tải lệnh
5. đưa WR là một xung âm.điều này bắt đầu lệnh xóa chip
6. đợi cho đến khi RDY/BSY lên cao trước khi tải một lệnh mới

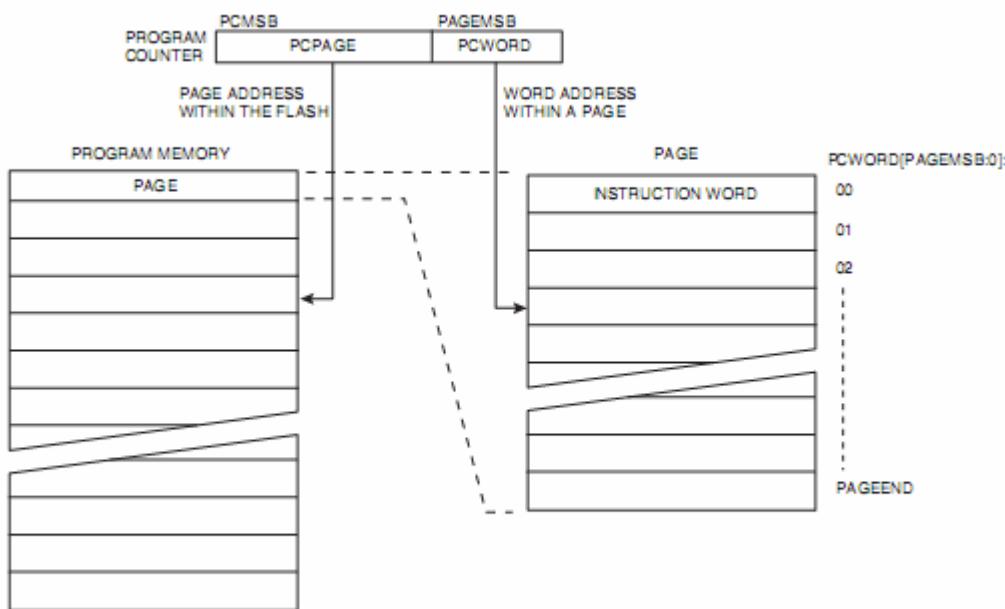
## Lập trình bộ nhớ flash

Bộ nhớ flash được tổ chức theo dạng trang,xem bảng 123 trang 291.khi lập trình bộ nhớ flash,dữ liệu chương trình được chia thành các trang.điều này cho phép 1 trang của dữ liệu chương trình được lập trình một cách đồng thời.các quy trình sau miêu tả cách để lập trình cho bộ nhớ flash:

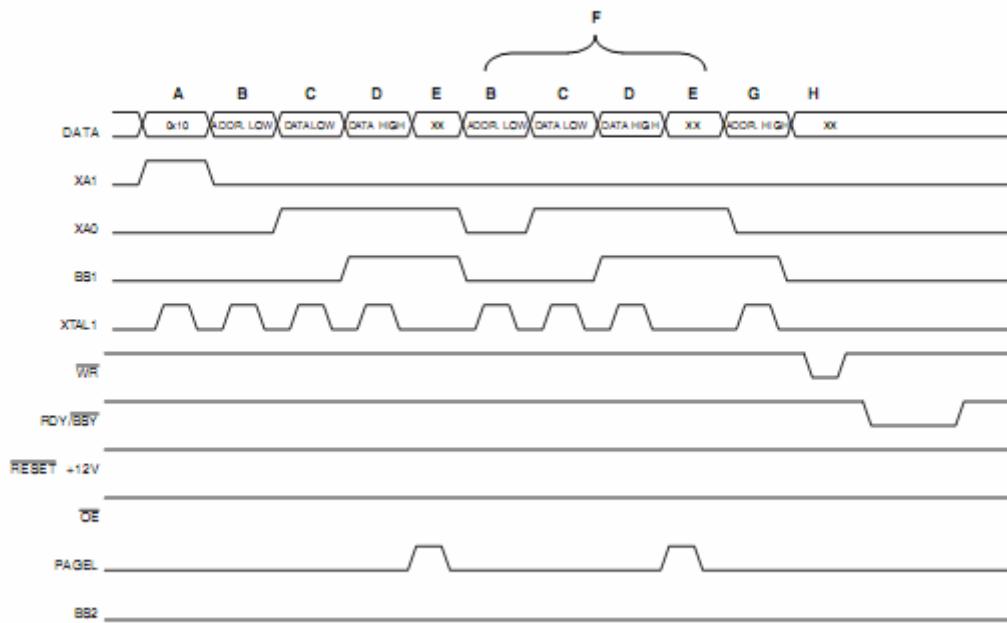
A.tải lệnh “viết flash”

- 1.đặt XA1,XA0 lên 10.điều này kích hoạt việc tải lệnh
  - 2.đặt BS1 lên 0
  - 3.đặt DATA lên 0001 0000.đây là lệnh cho việc viết flash
  - 4.đưa XTAL1 lên một sườn dương.điều này để tải lệnh
- B.tải địa chỉ byte thấp

- 1.đặt XA1,XA0 lên 00..điều này kích hoạt việc tải địa chỉ  
 2.đặt BS1 la 0..điều này lựa chọn địa chỉ thấp  
 3.đặt DATA bằng địa chỉ byte thấp(\$00-\$FF)địa chỉ thấp  
 4.đưa XTAL1 lên một sườn dương..điều này để tải các byte địa chỉ thấp  
 C.tải dữ liệu byte thấp  
 1.đặt XA1,XA0 lên 01..điều này kích hoạt việc tải dữ liệu byte thấp  
 2.đặt DATA bằng byte dữ liệu thấp  
 3.đưa XTAL1 lên một sườn dương..điều này để tải byte dữ liệu  
 D.tải byte dữ liệu cao  
 1.đặt BS1 lên 1..điều này lựa chọn byte dữ liệu cao  
 2.đặt XA1,XA0 lên 01..điều này kích hoạt việc tải byte dữ liệu cao  
 3.đặt DATA bằng byte dữ liệu cao.(\$00-\$FF)  
 4.đưa XTAL1 lên một sườn dương..điều này để tải byte dữ liệu  
 E.chốt dữ liệu  
 1.đặt BS1 lên 1..điều này lựa chọn byte dữ liệu cao  
 2.đưa PAGEL lên một sườn dương..điều này để chốt các byte địa chỉ.(xem hình 137về dạng các sóng tín hiệu)
- F.lập lại bước B cho đến bước E cho đến khi bộ đệm được điền đầy hoặc cho đến khi dữ liệu trong trang được tải.
- Trong khi các bit thấp trong địa chỉ được vẽ bẩn đồ lên các từ trong một trang,các bit địa chỉ cao hơn của trang trong bộ nhớ flash..điều này được minh họa trong hình 136 trang 294.chú ý rằng nếu một từ nhỏ hơn 8 bit là cần thiết để đánh địa chỉ từ trong trang(PAGE SIZE<256)cá bit có trọng số cao trong các byte có địa chỉ thấp được sử dụng để đánh địa chỉ cho trang khi tiến hành một quá trình viết trang
- G.tải địa chỉ byte cao  
 1.đặt XA1,XA0 lên 00..điều này kích hoạt việc tải địa chỉ.  
 2.đặt BS1 lên 1..điều này lựa chọn các địa chỉ mức cao  
 3.đặt DATA bằng địa chỉ byte cao (\$00-\$FF).  
 4.đưa XTAL1 lên một sườn dương..điều này tải các địa chỉ byte cao.
- H.lập trình trang  
 1.đặt BS1 =0  
 2.đưa WR lên một sườn âm..điều này bắt đầu lập trình dữ liệu của trang.RDY/BSY ở mức thấp.  
 3.đợi cho đến khi RDY/BSY lên cao(xem hình 137 để biết dạng tín hiệu)
- I.lập lại từ bước B đến bước H cho đến khi bộ nhớ được lập trình hoặc cho đến khi tất cả các dữ liệu được lập trình.
- J.kết thúc việc lập trình trang  
 1.cài XA1,XA0 lên 10..điều này kích hoạt việc tải lệnh.  
 2.đặt DATA lên 00000000..đây là lệnh cho không chế độ điều khiển.  
 3.đưa XTAL1 lên một sườn dương..điều này để tải lệnh,và các tín hiệu được viết bên trong được reset.

**Figure 136.** Addressing the Flash which is Organized in Pages

Note: 1. PCPAGE and PCWORD are listed in [Table 124 on page 292](#).

**Figure 137.** Programming the Flash Waveforms

Note: "XX" is don't care. The letters refer to the programming description above.

## Lập trình cho EEPROM

EEROM được tổ chức dưới dạng trang nhớ , xem bảng 124 trang 292 . Khi lập trình cho EEPROM , dữ liệu chương trình được chốt vào bên trong bộ đệm trang nhớ . Điều này cho phép 1 trang nhớ của dữ liệu được lập trình một cách đồng thời . Thuật

toán lập trình cho bộ nhớ dữ liệu EEPROM thì được theo như là ( tham khảo phần lập trình Flash trang 293 để thêm chi tiết về các lệnh , địa chỉ và việc tải dữ liệu ):

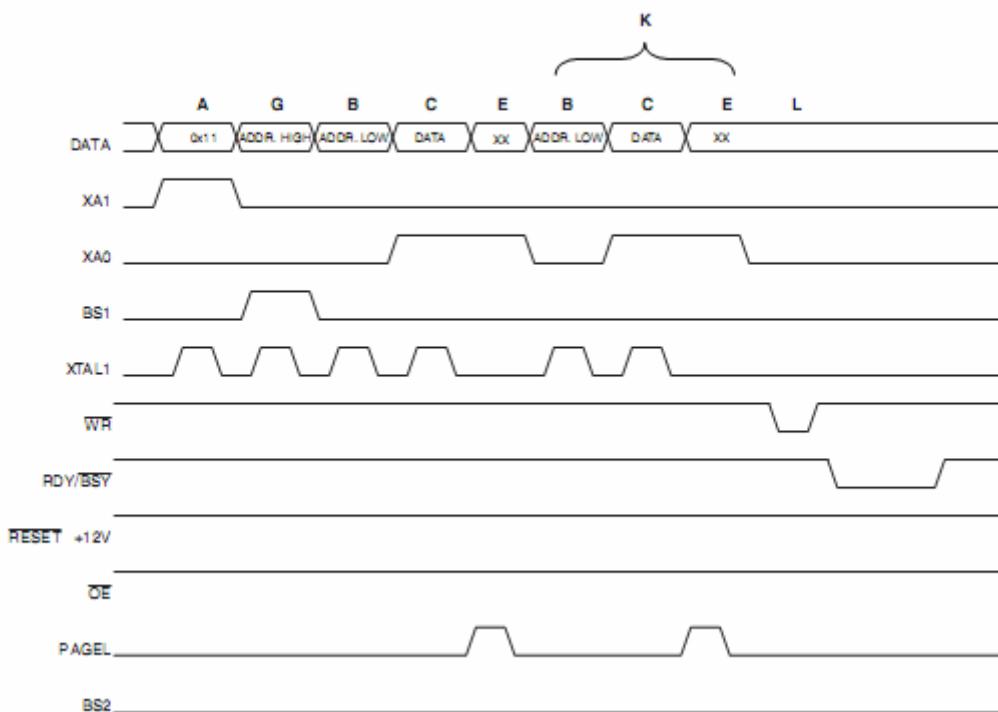
1. A : tải lệnh “0001 0001”
2. G : tải địa chỉ Byte cao (\$00 - \$FF)
3. B : tải địa chỉ Byte thấp (\$00 - \$FF)
4. C : tải dữ liệu (\$00 - \$FF)
5. E : chốt dữ liệu ( đưa PAGEL 1 xung dương )

K : lặp lại từ bước 3 đến bước 5 cho đến khi toàn bộ bộ đệm được điền đầy .

L : chương trình trang EEPROM

1. đặt BS1 là 0
2. đưa WR 1 xung âm . Điều này khởi động một quá trình lập trình của trang nhớ EEPROM . RDY/BSY đưa về mức thấp
3. Đợi cho đến khi RDY/BSY đến mức cao trước khi lập trình trang kế tiếp ( xem bảng 138 về dạng xung tín hiệu )

**Figure 138. Programming the EEPROM Waveforms**



## Đọc bộ nhớ Flash

Thuật toán để đọc bộ nhớ Flash được cho như bên dưới ( tham khảo phần lập trình Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A : tải lệnh “0000 0010 “
2. G : tải địa chỉ Byte cao (\$00 - \$FF)
3. B : tải địa chỉ Byte thấp (\$00 - \$FF)
4. đặt OE lên 0 và BS1 lên 0 . Các byte thấp từ Flash có thể đọc DATA
5. đặt BS1 lên 1 . Các byte cao của từ Flash có thể đọc DATA .

6. đặt OE lên 1

## **Đọc bộ nhớ EEPROM**

Thuật toán để đọc bộ nhớ EEPROM như là bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A : tải lệnh “0000 0011”
2. G : tải địa chỉ Byte cao (\$00 - \$FF)
3. B : tải địa chỉ Byte thấp (\$00 - \$FF)
4. đặt OE lên 0 và BS1 lên 0 . Byte dữ liệu EEPROM có thể được đọc tại DATA
5. đặt OE lên 1

## **Lập trình các bit thấp cầu chì**

Thuật toán để lập trình cho các bit thấp cầu chì như bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A : tải lệnh “0100 0000”
2. C : tải byte dữ liệu thấp . Bit n = 0 và bit n = 1 xóa các bit cầu chì .
3. đặt BS1 lên 0 và BS2 lên 0
4. đưa vào WR 1 xung âm và đợi cho RDY/BSY lên cao

## **Lập trình các bit cầu chì cao**

Thuật toán cho việc lập trình các bit cầu chì cao như bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

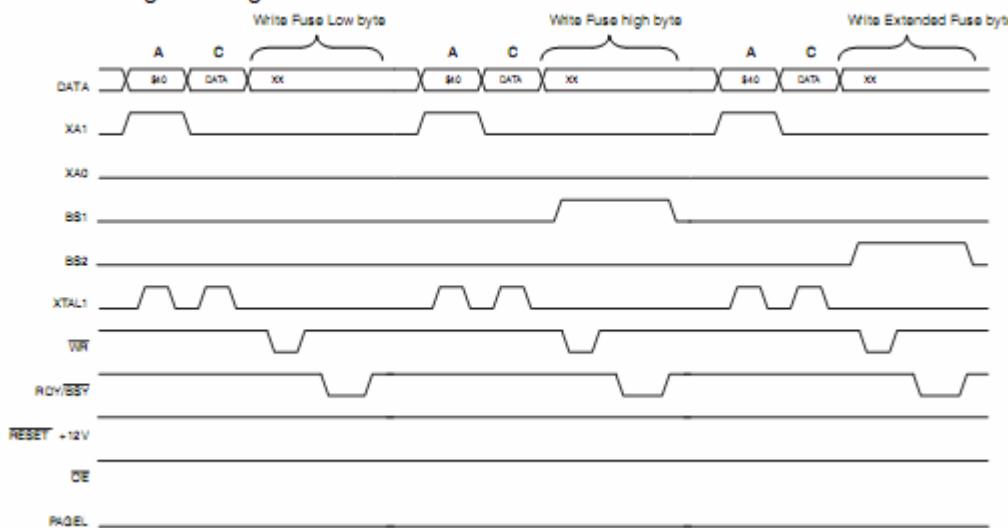
1. A tải lệnh ‘0100 0000’
2. C : tải byte dữ liệu thấp . Bit n = 0 chương trình và bit n = 1 xóa các bit cầu chì .
3. Đặt BS1 lên 1 và BS2 lên 0 . Điều này lựa chọn byte dữ liệu cao .
4. đưa vào WR một xung âm và đợi cho đến khi đợi cho RDY/BSY lên cao
5. đặt BS1 lên 0 . Điều này lựa chọn Byte dữ liệu thấp

## **Lập trình các byte cầu chì mở rộng**

Thuật toán cho việc lập trình các bit cầu chì mở rộng thì như bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A : tải lệnh “0100 0000”
2. C : tải byte dữ liệu thấp . Bit n = 0 và bit n = 1 xóa bit cầu chì

3. Đặt BS1 lên 1 và BS2 lên 0 . Điều này lựa chọn byte dữ liệu cao .
4. đưa vào WR một xung âm và đợi cho đến khi RDY/BSY lên cao
5. đặt BS2 lên 0 . Điều này lựa chọn byte dữ liệu .

**Figure 139. Programming the Fuses**

## Lập trình các bit khóa

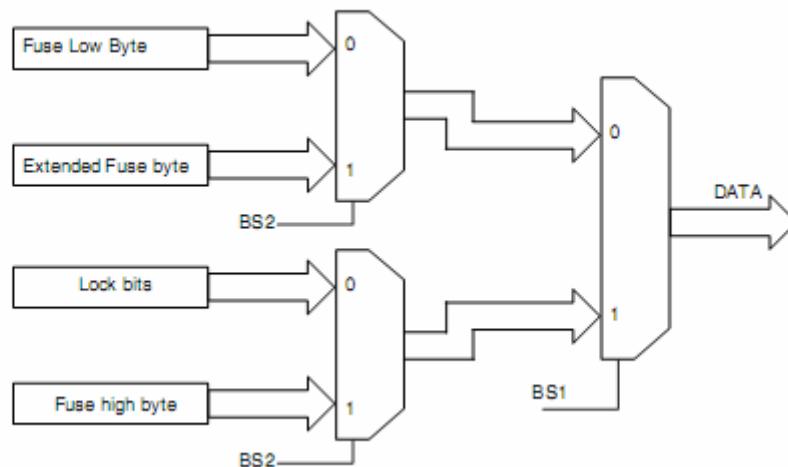
Thuật toán để lập trình cho các bit khóa như bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A : tải lệnh “0010 0000”
2. C : tải các byte dữ liệu thấp . Bit n = 0 lập trình các bit khóa .
3. đưa vào WR một xung âm và đợi cho đến khi RDY/BSY lên cao các bit khóa chỉ có thể bị xóa bằng việc thực hiện lệnh xóa chip

## Đọc các bit khóa và bit cầu chì

Thuật toán cho việc đọc các bit cầu chì và các bit khóa như sau (( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A : tải lệnh 0000 0100
2. đặt OE lên 0 và BS2 lên 0 , và BS1 lên 0 . Trạng thái của các bit thấp cầu chì có thể được đọc bây giờ tại DATA ( 0 nghĩa là đã lập trình )
3. đặt OE lên 0 , BS2 lên 1 , và BS1 lên 1 . Trạng thái của các bit cầu chì cao có thể được đọc bây giờ tại DATA ( 0 nghĩa là đã lập trình )
4. đặt OE lên 0 , BS2 lên 1 , và BS1 lên 0 . Trạng thái của các bit cầu chì mở rộng có thể được đọc tại DATA ( 0 nghĩa là đã lập trình )
5. đặt OE lên 0 , BS2 lên 0 , và BS1 lên 1 . Trạng thái của các bit khóa có thể được đọc tại DATA ( 0 nghĩa là đã lập trình 0 )
6. Đặt OE là 1

**Figure 140.** Mapping Between BS1, BS2 and the Fuse- and Lock Bits During Read

### Đọc các byte kí hiệu

Thuật toán cho việc đọc các byte kí hiệu thì như bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

1. A tải lệnh 0000 1000
2. B : tải các địa chỉ Byte thấp (\$00 - \$02)
3. đặt OE lên 0 và BS1 lên 0 . Các byte kí hiệu đã lựa chọn có thể được đọc tại DATA
4. đặt OE lên 1

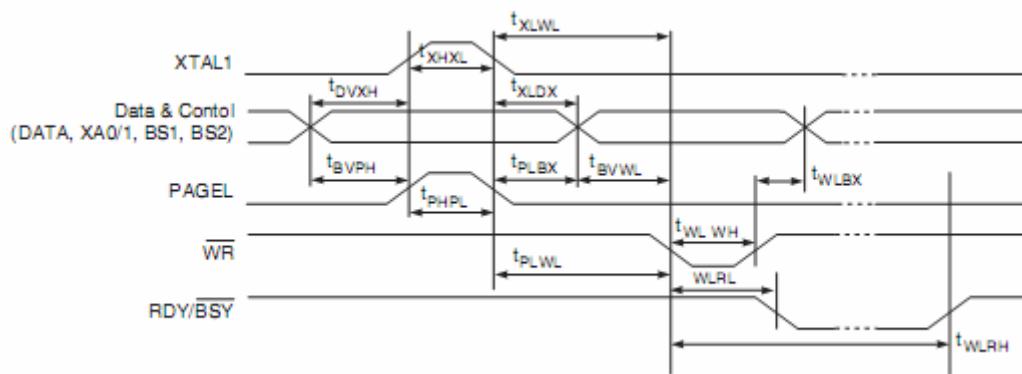
### Đọc byte hiệu chỉnh

Thuật toán dùng để đọc các byte hiệu chỉnh như bên dưới ( tham khảo phần lập trình bộ nhớ Flash trên trang 293 để biết thêm chi tiết về quá trình tải lệnh và địa chỉ )

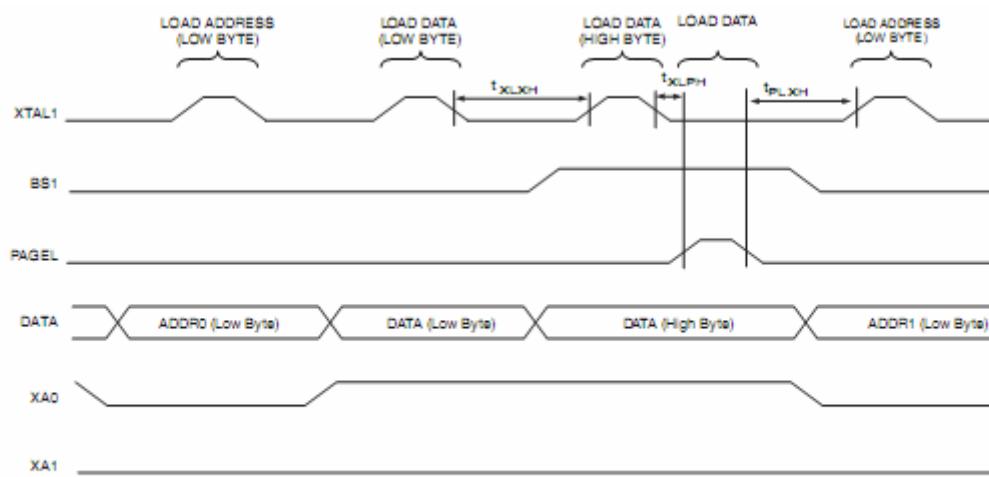
1. A : tải lệnh “0000 1000 “
2. B : tải các byte địa chỉ thấp
3. đặt OE là 0 và BS1 là 1 . Byte hiệu chỉnh có thể được đọc tại DATA
4. đặt OE lên 1

### Các thông số lập trình song song

Hình 141 . giản đồ thời gian lập trình song song , bao gồm bộ định thời chung cần thiết

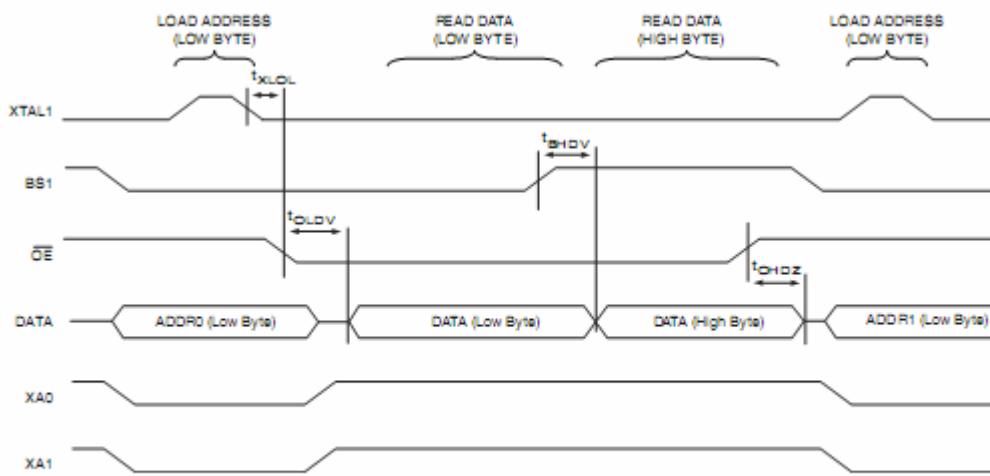


Hình 142 . gián đồ thời gian lập trình song song , quá trình tải liên tiếp với các yêu cầu định thời



Note: The timing requirements shown in Figure 141 (i.e.  $t_{DVXH}$ ,  $t_{XHXL}$ , and  $t_{XLDX}$ ) also apply to loading operation.

Hình 143 . gián đồ thời gian của lập trình song song , việc đọc liên tiếp (trong trang giống nhau ) với đòi hỏi định thời .



Note: The timing requirements shown in Figure 141 (i.e.  $t_{DVXH}$ ,  $t_{XHXL}$ , and  $t_{XLDX}$ ) also apply to reading operation.

Bảng 126 . Các thông số kĩ thuật của lập trình song song ,  $V_{CC} = 5V \pm 10\%$

Symbol	Parameter	Min	Typ	Max	Units
$V_{PP}$	Programming Enable Voltage	11.5		12.5	V
$I_{PP}$	Programming Enable Current			250	$\mu A$
$t_{DVXH}$	Data and Control Valid before XTAL1 High	67			ns
$t_{XHXL}$	XTAL1 Low to XTAL1 High	200			ns
$t_{XHXL}$	XTAL1 Pulse Width High	150			ns
$t_{XLDX}$	Data and Control Hold after XTAL1 Low	67			ns
$t_{XLWL}$	XTAL1 Low to WR Low	0			ns

Symbol	Parameter	Min	Typ	Max	Units
$t_{XLPH}$	XTAL1 Low to PAGEL high	0			ns
$t_{PLXH}$	PAGEL low to XTAL1 high	150			ns
$t_{BVPH}$	BS1 Valid before PAGEL High	67			ns
$t_{PHPL}$	PAGEL Pulse Width High	150			ns
$t_{PLBX}$	BS1 Hold after PAGEL Low	67			ns
$t_{WLBX}$	BS2/1 Hold after WR Low	67			ns
$t_{PLWL}$	PAGEL Low to WR Low	67			ns
$t_{BVWL}$	BS1 Valid to WR Low	67			ns
$t_{WLWH}$	WR Pulse Width Low	150			ns
$t_{WLRL}$	WR Low to RDY/BSY Low	0		1	$\mu s$
$t_{WLRH}$	WR Low to RDY/BSY High <sup>(1)</sup>	3.7		5	ms
$t_{WLRH\_CE}$	WR Low to RDY/BSY High for Chip Erase <sup>(2)</sup>	7.5		10	ms
$t_{XLOL}$	XTAL1 Low to OE Low	0			ns
$t_{BVDV}$	BS1 Valid to DATA valid	0		250	ns
$t_{OLDV}$	OE Low to DATA Valid			250	ns
$t_{OHDZ}$	OE High to DATA Tri-stated			250	ns

Notes: 1.  $t_{WLRH}$  is valid for the Write Flash, Write EEPROM, Write Fuse bits and Write Lock bits commands.  
 2.  $t_{WLRH\_CE}$  is valid for the Chip Erase command.

## Quá trình tải nối tiếp

Cả hai bộ nhớ Flash và EEPROM có thể được lập trình sử dụng bus SPI nối tiếp trong khi RESET được kéo vào GND . Giao diện nối tiếp thì bao gồm các chân của SCK, MOSI (đầu vào) và MISO (đầu ra) . Sau khi RESET ở mức thấp , lệnh kích hoạt lập trình cần được thực thi trước tiên trước khi hoạt động lập trình/xóa có thể được thực thi . Chú ý rằng, trong bảng 127 trang 300 , sự vẽ bản đồ chân cho lập trình SPI được liệt kê . Không phải tất cả các phần của các chân SPI đều phục vụ cho giao diện SPI bên trong . Chú ý rằng thông qua việc miêu tả về việc tải nối tiếp , MOSI và MISO được sử dụng để miêu tả dữ liệu nối tiếp vào và dữ liệu nối tiếp ra một cách tương ứng . Với Atmega 128 , các chân này được maped lên PDI và PDO .

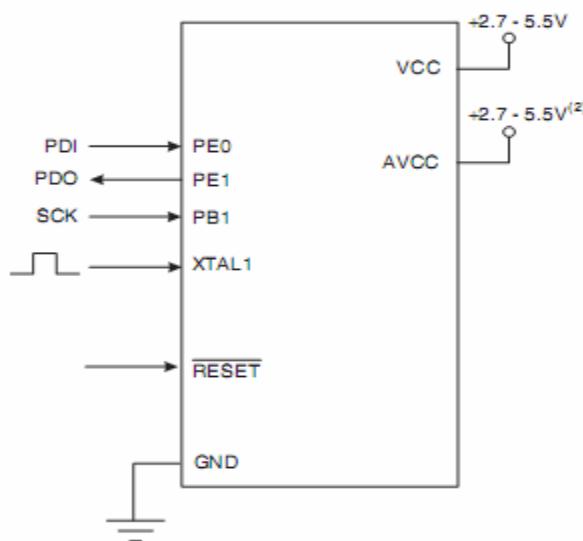
## Quá trình vẽ bản đồ chân lập trình nối tiếp SPI

Mặc dù giao diện lập trình SPI sử dụng trước (re – uses) các module I/O SPI , có một điểm khác biệt quan trọng : các chân MOSI/MISO mà được vẽ bản đồ lên PB2 và PB3 trong module I/O của SPI thì không được sử dụng trong giao diện lập trình . thay vì , PE0 và PE1 được sử dụng cho dữ liệu trong chế độ lập trình SPI như được chỉ ra trong hình 127

Table 127. Pin Mapping SPI Serial Programming

Symbol	Pins	I/O	Description
MOSI (PDI)	PE0	I	Serial data in
MISO (PDO)	PE1	O	Serial data out
SCK	PB1	I	Serial clock

Figure 144. SPI Serial Programming and Verify<sup>(1)</sup>



- Notes:
- If the device is clocked by the Internal Oscillator, it is no need to connect a clock source to the XTAL1 pin.
  - $V_{CC} - 0.3V < AVCC < V_{CC} + 0.3V$ , however, AVCC should always be within 2.7 - 5.5V.

Khi lập trình EEPROM , 1 chu kì tự động xóa được xây dựng bên trong quá trình lập trình tự định thời (soft-timed) ( chỉ trong chế độ nối tiếp ) và không cần thiết để thực thi đầu tiên lệnh xóa chip . Quá trình xóa chip chuyển hướng thành phần của các vùng nhớ trong cả hai bộ nhớ EEPROM trong \$FF

Phụ thuộc vào các cầu chì CKSEL , 1 xung nhịp có hiệu lực phải được đưa ra . Chu kì thấp và cao nhất chodầu vào xung nhịp (SCK ) được xác định như dưới :

Thấp : > 2 chu kì xung nhịp CPU cho  $f_{CK} < 12MHz$  , 3 chu kì xung nhịp cho  $f_{CK} < 12MHz$

Cao : > 2 chu kì xung nhịp CPU cho  $f_{CK} < 12MHz$  , 3 chu kì xung nhịp cho  $f_{CK} < 12MHz$

## Thuật toán lập trình nối tiếp SPI

Khi viết dữ liệu nối tiếp lên Atmega 128 , dữ liệu bị xóa trên sườn lên của xung SCK

Khi đọc dữ liệu từ Atmega 128 , dữ liệu bị khóa trên sườn xuống của SCK . xem hình 145 về chi tiết bộ định thời .

Để lập trình và kiểm tra lại Atmega 128 trong chế độ lập trình nối tiếp SPI , các chuỗi kế tiếp sau đây được khuyến cáo ( xem các dạng lệnh 4 byte trên bảng 145)

### 1. chuỗi cấp điện ( Power – up sequence )

đặt nguồn điện giữa  $V_{CC}$  và GND trong khi chân RESET và SCK đặt là 0 . Trong vài hệ thống , người lập trình không thể đảm bảo rằng SCK được giữ ở mức thấp trong suốt quá trình cấp điện . Trong trường hợp này , RESET phải được đưa vào 1 xung dương trong khoảng dưới 2 chu kì xung nhịp CPU sau khi SCK vừa được đặt là 0

Như một sự lựa chọn để sử dụng tín hiệu RESET , chân PEN có thể được giữ thấp trong suốt quá trình Reset bật nguồn trong khi SCK đặt là 0 . Trong trường hợp này , chỉ có giá trị của PEN tại reset bật nguồn là quan trọng . Nếu người lập trình không thể đảm bảo rằng SCK được giữ mức thấp trong suốt quá trình cấp điện (power – up) , phương pháp PEN có thể được sử dụng . Thiết bị phải được ngắt điện theo thứ tự bắt đầu quá trình hoạt động bình thường khi sử dụng phương pháp này .

2 . Đợi trong khoảng dưới 20ms và kích hoạt lập trình nối tiếp SPI bằng việc gửi lệnh lập trình kích hoạt nối tiếp lên chân MOSI

3. Các lệnh lập trình nối tiếp SPI sẽ không làm việc nếu như quá trình truyền thông ra ngoài quá trình đồng bộ hóa . Khi trong chế độ sync byte thứ 2 (\$53), sẽ phản hồi lại khi sự đưa ra byte thứ 3 của lệnh kích hoạt lập trình . Lựa chọn phản hồi là đúng hoặc sai , tất cả 4 byte của lệnh phải được được chuyển phát . Nếu \$53 không phản hồi lại , đưa chân RESET 1 xung dương và đưa ra 1 lệnh kích hoạt lập trình mới .

4. Bộ nhớ Flash được lập trình mỗi trang nhớ tại một thời điểm . Cỡ của trang thì được tìm thấy trong bảng 124 trên trang 292 . Trang nhớ thì được tải một byte tại 1 thời điểm bằng việc cung cấp 7 LSB của địa chỉ và dữ liệu cùng với lệnh tải chương trình trang nhớ. Để đảm bảo quá trình tải đúng các trang nhớ , các byte dữ liệu thấp phải được tải trước byte dữ liệu cao được áp dụng cho các địa chỉ được đưa ra . Các trang nhớ chương trình được lưu trữ bằng việc tải lệnh viết chương trình trang nhớ với 9 MSB của địa chỉ . Nếu quá trình hỏi vòng không được sử dụng, người sử dụng phải đợi trong khoảng thời gian  $t_{WD\_FLASH}$  trước khi đưa ra trang mới. (xem bảng 128)

Chú ý : nếu các lệnh khác hơn quá trình hỏi vòng được đặt trước bất cứ quá trình viết nào (Flash , EEPROM , các bit khóa , Cầu chì ) được hoàn thành , có thể gây ra sự lặp trình không đúng .

5. Mảng nhớ EEPROM thì được lập trình một byte tại một thời điểm bằng việc cấp địa chỉ và dữ liệu cùng với lệnh viết được chấp thuận . 1 vùng nhớ EEPROM được xóa trước tiên một cách tự động trước khi một dữ liệu mới được viết . Nếu sự hỏi vòng không được sử dụng , người sử dụng phải đợi trong khoảng  $t_{WD\_EEPROM}$  trước khi xuất ra byte trước . ( xem bảng 128) . Trong một thiết bị được xóa , không có \$FF trong các file dữ liệu cần thiết được lập trình .
6. bất cứ vùng nhớ nào có thể được kiểm tra bằng việc sử dụng các lệnh đọc , cái mà phản hồi lại thành phần tại địa chỉ được lựa chọn tại các đầu ra nối tiếp MISO
7. tại phần kết thúc của phần lập trình , RESET có thể được cài đặt ở mức cao bắt đầu hoạt động bình thường
8. Power –OFF kế tiếp ( nếu cần thiết )  
Đặt chân RESET là 1  
Chuyển V<sub>CC</sub> sang tắt nguồn

## Sự hỏi vòng Flash

Khi một trang nhớ đang được lập trình vào trong bộ nhớ Flash , việc đọc một vùng địa chỉ trong trang đang được lập trình sẽ đưa ra giá trị \$FF . Tại thời điểm mà thiết bị sẵn sàng cho 1 trang nhớ mới , giá trị đã lập trình sẽ được đọc một cách chính xác . Điều này thường được xác định khi trang kế tiếp có thể được viết . Chú ý rằng toàn bộ trang nhớ được viết 1 cách đồng thời và bất cứ địa chỉ nào trong trang có thể được sử dụng cho sự hỏi vòng . Sự hỏi vòng dữ liệu của bộ nhớ Flash sẽ không làm việc với giá trị \$FF , vì vậy khi lập trình giá trị này , người sử dụng sẽ phải đợi trong khoảng thời gian  $t_{WD\_FLASH}$  trước khi lập trình trang kế tiếp . Như là một thiết bị xóa chip bao gồm \$FF trong tất cả các vùng địa chỉ , sự lập trình của địa chỉ mà bao gồm \$FF , có thể bị giữ lại . Xem bảng 128 về giá trị của  $t_{WD\_FLASH}$

## Sự hỏi vòng dữ liệu EEPROM

Khi một byte mới vừa được viết và đang được lập trình vào trong EEPROM , việc đọc các vùng địa chỉ đang được lập trình sẽ đưa ra giá trị \$FF . Tại thời điểm mà thiết bị sẵn sàng được lập trình cho 1 byte mới , giá trị được lập trình sẽ được đọc một cách chính xác . Điều này thường được xác định khi byte kế tiếp có thể đọc . Điều này sẽ không làm việc cho giá trị \$FF trong tất cả các vùng nhớ , nhưng người sử dụng nên có các ý tưởng sau : Như là một thiết bị xóa chíp bao gồm \$FF trong tất cả các vùng nhớ , việc lập trình cho các địa chỉ này bao gồm \$FF , có thể được giữ . Điều này không áp dụng nếu như EEPROM được lập trình lại mà thiếu thiết bị xóa chíp . Trong trường hợp này , quá trình hỏi vòng giữ liệu không thể được sử dụng cho giá trị \$FF , và người sử dụng sẽ phải đợi trong khoảng thời gian  $t_{WD\_EEPROM}$  trước khi lập trình byte kế tiếp . Xem bảng 128, để biết giá trị  $t_{WD\_EEPROM}$

**Table 128.** Minimum Wait Delay before Writing the Next Flash or EEPROM Location,  $V_{CC} = 5 V \pm 10\%$

Symbol	Minimum Wait Delay
$t_{WD\_FUSE}$	4.5 ms
$t_{WD\_FLASH}$	5 ms
$t_{WD\_EEPROM}$	10 ms
$t_{WD\_ERASE}$	10 ms

**Figure 145.** SPI Serial Programming Waveforms

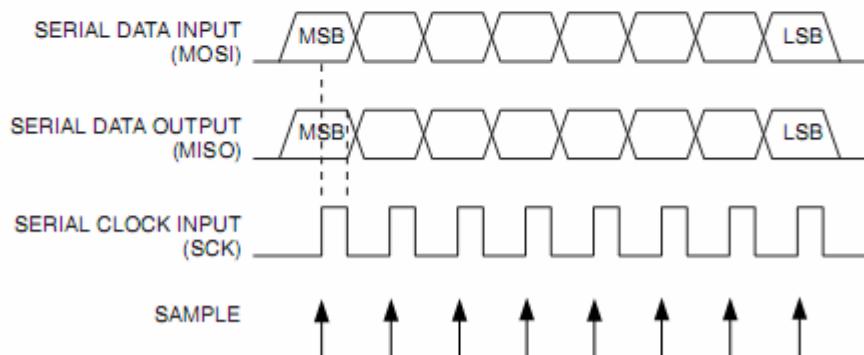


Table 129. SPI Serial Programming Instruction Set

Instruction	Instruction Format				Operation
	Byte 1	Byte 2	Byte 3	Byte4	
Programming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx	Enable SPI Serial Programming after RESET goes low.
Chip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase EEPROM and Flash.
Read Program Memory	0010 H000	aaaa aaaa	bbbb bbbb	oooo oooo	Read H (high or low) data o from Program memory at word address a:b.
Load Program Memory Page	0100 H000	xxxx xxxx	xbbb bbbb		Write H (high or low) data i to Program Memory page at word address b. Data low byte must be loaded before data high byte is applied within the same address.
Write Program Memory Page	0100 1100	aaaa aaaa	bxxx xxxx	xxxx xxxx	Write Program Memory Page at address a:b.
Read EEPROM Memory	1010 0000	xxxx aaaa	bbbb bbbb	oooo oooo	Read data o from EEPROM memory at address a:b.
Write EEPROM Memory	1100 0000	xxxx aaaa	bbbb bbbb		Write data i to EEPROM memory at address a:b.
Read Lock bits	0101 1100	0000 0000	xxxx xxxx	xxoo oooo	Read Lock bits. "0" = programmed, "1" = unprogrammed. See <a href="#">Table 115 on page 286</a> for details.
Write Lock bits	1010 1100	111x xxxx	xxxx xxxx	11	Write Lock bits. Set bits = "0" to program Lock bits. See <a href="#">Table 115 on page 286</a> for details.
Read Signature Byte	0011 0000	xxxx xxxx	xxxx xxbb	oooo oooo	Read Signature Byte o at address b.
Write Fuse bits	1010 1100	1010 0000	xxxx xxxx		Set bits = "0" to program, "1" to unprogram. See <a href="#">Table 119 on page 286</a> for details.
Write Fuse High Bits	1010 1100	1010 1000	xxxx xxxx		Set bits = "0" to program, "1" to unprogram. See <a href="#">Table 118 on page 286</a> for details.
Write Extended Fuse bits	1010 1100	1010 0100	xxxx xxxx	xxxx xxll	Set bits = "0" to program, "1" to unprogram. See <a href="#">Table 119 on page 286</a> for details.
Read Fuse bits	0101 0000	0000 0000	xxxx xxxx	oooo oooo	Read Fuse bits. "0" = programmed, "1" = unprogrammed. See <a href="#">Table 119 on page 286</a> for details.
Read Extended Fuse bits	0101 0000	0000 1000	xxxx xxxx	oooo oooo	Read Extended Fuse bits. "0" = programmed, "1" = unprogrammed. See <a href="#">Table 119 on page 286</a> for details.
Read Fuse High Bits	0101 1000	0000 1000	xxxx xxxx	oooo oooo	Read Fuse high bits. "0" = programmed, "1" = unprogrammed. See <a href="#">Table 118 on page 286</a> for details.
Read Calibration Byte	0011 1000	xxxx xxxx	0000 00bb	oooo oooo	Read Calibration Byte o at address b.

Note:  
 a = address high bits  
 b = address low bits  
 H = 0 - Low byte, 1 - High Byte  
 o = data out  
 i = data in  
 x = don't care

## Thông số kỹ thuật của lập trình SPI

Về thông số kỹ thuật của modun SPI, xem thông số kỹ thuật giản đồ thời gian SPI.

## Lập trình thông qua giao diện JTAG..

Lập trình thông qua giao diện JTAG cần thiết để điều khiển bốn chân xác định JTAG: TCK, TMS, TDI, TDO. Quá trình điều khiển của chân reset và các chân đồng hồ là không cần thiết.

Có thể sử dụng giao diện JTAG, cầu chì JTAGEN phải được lập trình. Thiết bị này được nén mặc định với cầu chì đã được lập trình. Thêm vào đó, bit JTD trong thanh ghi MCUCSR phải được xóa. Như một sự lựa chọn, nếu bit JTD được cài đặt, reset ngoài có thể đặt ở mức thấp. Sau đó bit JTD có thể được xóa, sau 2 chu kỳ xung nhịp của chip, và các chân JTAG thì khả dụng trong chế độ lập trình. Điều này cung cấp một khả năng để sử dụng chân JTAG như là các công thông thường trong chế độ đang chạy trong khi

vẫn cho phép lập trình trong hệ thống thông qua giao diện JTAG. Chú ý rằng kỹ thuật này có thể không được sử dụng khi dùng các chân JTAG cho chế độ quét thứ cấp và chế độ hiệu chỉnh lỗi trên chip. Trong các trường hợp này các chân JTAG phải được sử dụng cho các chức năng này.

Như là được xác định trong tài liệu này, LSB được nén vào trong và ra ngoài đầu tiên của thanh ghi Shift.

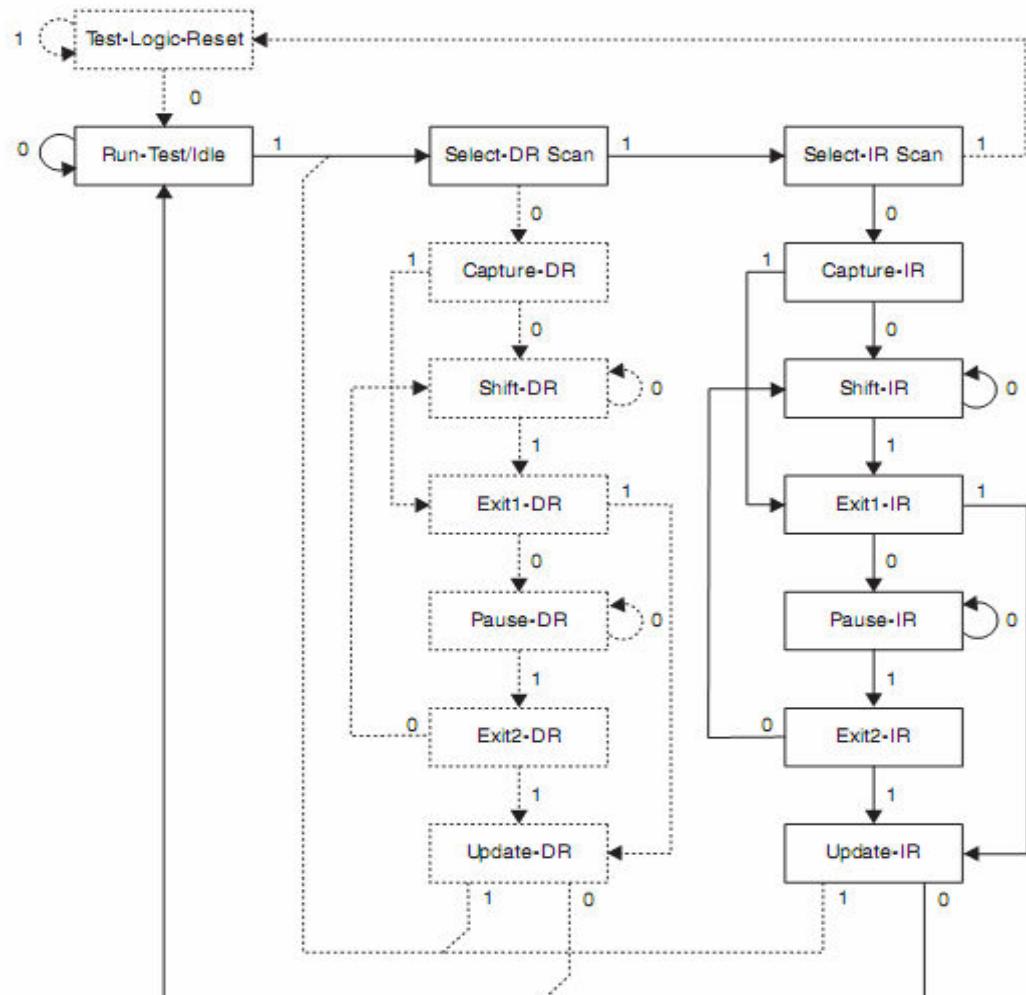
### Các lệnh lập trình xác định JTAG.

Thanh ghi các lệnh thì có độ rộng là 4 bit, được hỗ trợ lên đến 16 lệnh. Các lệnh JTAG hữu dụng thì được liệt kê bên dưới.

Bit OPCODE cho mỗi lệnh được chỉ ra đằng sau các lệnh trong định dạng hex. Sự miêu tả các thanh ghi dữ liệu được lựa chọn như là một phần giữa các bit TDI và TDO cho mỗi lệnh.

Trạng thái Run-Test/idle của bộ điều khiển TAP được sử dụng để phát ra các xung nhịp bên trong. Nó cũng có thể được sử dụng như trạng thái IDLE giữa các JTAG nối tiếp. Các máy phát trạng thái kế tiếp cho sự thay đổi các từ lệnh được chỉ ra trong hình 146.

**Figure 146.** State Machine Sequence for Changing the Instruction Word



## **AVR\_RESET (\$C).**

AVR xác định các lệnh JTAG chung cho việc cài đặt các thiết bị AVR trong chế độ reset hoặc đưa các thiết bị ra khỏi chế độ reset. Bộ điều khiển TAP thì không được reset bằng lệnh này. Một thanh ghi reset được lựa chọn như là một thanh ghi dữ liệu. Chú ý rằng lệnh reset này sẽ hoạt động chỉ cần có một mức logic 1 trên chân RESET chain. Đầu ra từ chain này thì không bị chốt.

Các trạng thái hoạt động là:

- Shift-DR: Thanh ghi reset được Shift bởi đầu vào TCK.

## **PROG\_ENABLE (\$4)**

AVR xác định lệnh JTAG chung để kích hoạt lập trình thông qua cổng JTAG. Thanh ghi kích hoạt lập trình 16 bit được lựa chọn như là thanh ghi dữ liệu. Các trạng thái hoạt động thì được chỉ ra như bên dưới:

- Shift-DR: Tín hiệu kích hoạt lập trình được Shift bên trong thanh ghi dữ liệu.
- Update DR: Tín hiệu kích hoạt lập trình được so sánh với giá trị chuẩn, và chế độ lập trình được truy nhập nếu như tín hiệu của nó có hiệu lực.

## **PROG\_COMMANDS (\$5)**

AVR xác định các lệnh JTAG để truy nhập các lệnh lập trình thông qua cổng JTAG. Thanh ghi lệnh lập trình 15 bit được lựa chọn như là thanh ghi dữ liệu. Các trạng thái hoạt động được liệt kê như bên dưới:

- Capture-DR: Kết quả của lệnh trước được tải vào thanh ghi dữ liệu
- shift DR: Thanh ghi dữ liệu được Shift bởi đầu vào TCK, được nén vào kết quả của lệnh trước và nén vào lệnh mới
- Update-DR: Lệnh lập trình được đặt vào các đầu vào flash
- Run-Test/idle: Một chu kỳ xung nhịp được phát ra, đang thực thi một lệnh được áp dụng.

## **PROG\_PAGELOAD (\$6)**

AVR xác định các lệnh JTAG chung để tải trực tiếp trang dữ liệu flash thông qua cổng JTAG. Thanh ghi tải trang flash ảo bit 2048 được lựa chọn như là một thanh ghi dữ liệu. Đây là một vòng quét ảo với độ dài bằng số thứ tự của các bit trong trang flash. Bên trong thanh ghi shift là 8 bit. Không giống như tất cả các lệnh trong JTAG, trạng thái Capture-DR không được sử dụng để chuyển dữ liệu vào thanh ghi shift. Dữ liệu tự động được chuyển từ các bite bộ đệm trang flash bằng bite trong trạng thái shift DR bằng một máy tạo xung trạng thái bên trong. Đây chỉ là các trạng thái:

- Shift-DR: Dữ liệu flash được nén vào trong từ TDI bằng đầu vào TCK, và tự động được tải vào trong trang flash mỗi bite 1 lần.

Chú ý: Lệnh JTAG,PROG\_PAGELOAD (\$6) chỉ có thể được sử dụng nếu thiết bị AVG là thiết bị đầu tiên trong chuỗi quét JTAG. Nếu như AVG không thể là thiết bị đầu tiên trong chuỗi quét, bite-wise thuật toán lập trình phải được sử dụng.

## **PROG\_PAGEREAD (\$7).**

-AVR xác định các lệnh JTAG chung để đọc một trang dữ liệu flash đầy thông qua cổng JTAG. Thanh ghi đọc trang flash ảo 2056 bite được lựa chọn như một thanh ghi trang thái. Đây là mmotj chuỗi quét ảo với độ dài bằng số lượng các bit trong một trang flash. Bên trong thanh ghi shift là 8 bit. Không giống như hầu hết các lệnh JTAG khác, trạng thái capture-DR không đuwocj sử dụng để chuyển dữ liệu vào thanh ghi shift. Dữ liệu thì được tự động chuyển từ bộ đệm trang flash bằng bite trong trạng thái shift-DR bằng một máy phát trạng thái bên trong. Đây chỉ là các trạng thái haotj động:

- Shift-DR: Dữ liệu flash được tự động đọc mỗi bite một lần và được nén ra ngoài trên chân TDO bằng đầu vào TCK. Đầu vào TDI được bỏ qua.

## **Các thanh ghi dữ liệu**

Các thanh ghi dữ liệu được lựa chọn bằng các thanh ghi lệnh JTAG được miêu tả ở trang 305. Các thanh ghi dữ liệu hữu ích cho hoạt động lập trình là:

- thanh ghi reset
- Thanh ghi kích hoạt lập trình
- Thanh ghi lệnh lập trình
- Thanh ghi tải trang flash ảo
- Thanh ghi đọc trang flash ảo

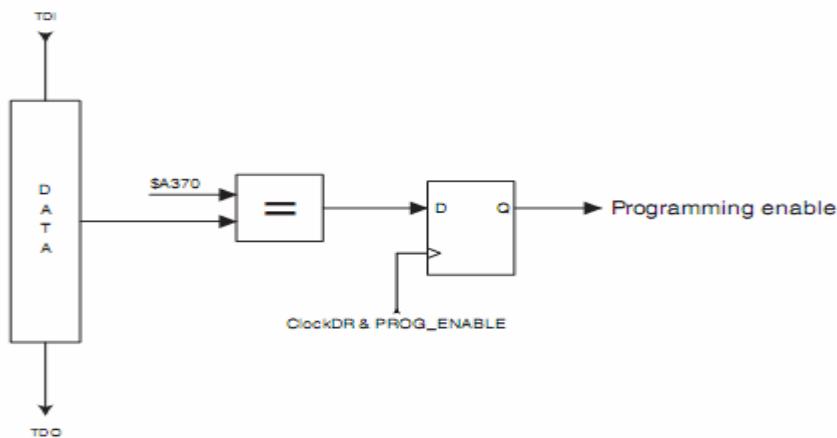
## **Thanh ghi reset**

- Thanh ghi reset là một thanh ghi kiểm tra dữ liệu được sử dụng để reset một bộ phận trong suốt quá trình lập trình. Nó thì cần thiết để reset một bộ phận trước khi truy nhập vào chế độ lập trình.
- Một giá trị cao trong thanh ghi reset tương ứng để kéo reset bên ngoài xuống mức thấp. Phần này được reset chỉ cần có một giá trị cao đưa ra ở trong thanh ghi reset. Sự phụ thuộc vào sự cài đặt bit cầu chì cho sự lựa chọn xung nhịp, bộ phận còn lại sẽ reset cho một chu kỳ reset time out (Tham khảo trang 37 ) sau khi giải phóng thanh ghi reset. Đầu ra từ thanh ghi dữ liệu này không bị chốt, vì vậy quá trình reset sẽ xảy ra ngay lập tức như được chỉ ra trong hình 123 trang 254.

## Thanh ghi kích hoạt chương trình.

Thanh ghi kích hoạt chương trình là một thanh ghi 16 bit. Thành phần của thanh ghi này được so sánh với tín hiệu kích hoạt lập trình, mã nhị phân 1010\_0011\_0111\_0000. Khi các thành phần của thanh ghi này bằng với tín hiệu kích hoạt lập trình, lập trình thông qua cổng JTAG được kích hoạt. Thanh ghi được reset về không trong chế độ reset power on, và nên luôn luôn được reset khi rời khỏi chế độ lập trình.

Figure 147. Programming Enable Register



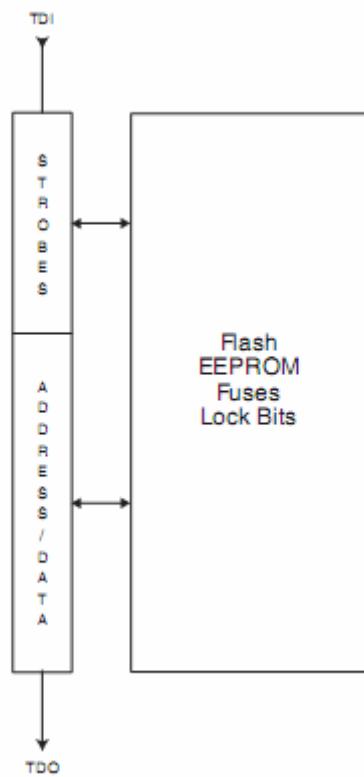
## Thanh ghi lệnh lập trình

Thanh ghi lệnh lập trình là một thanh ghi 15 bit. Thành ghi này được sử dụng để nén liên tiếp trong các lệnh lập trình, và được nén ra ngoài liên tiếp cho kết quả của các lệnh phía trước. Lệnh lập trình JTAG được cài đặt như được chỉ ra trong bảng 130. Trạng thái kế tiếp khi đang shift các lệnh lập trình được minh họa trong các hình 149.

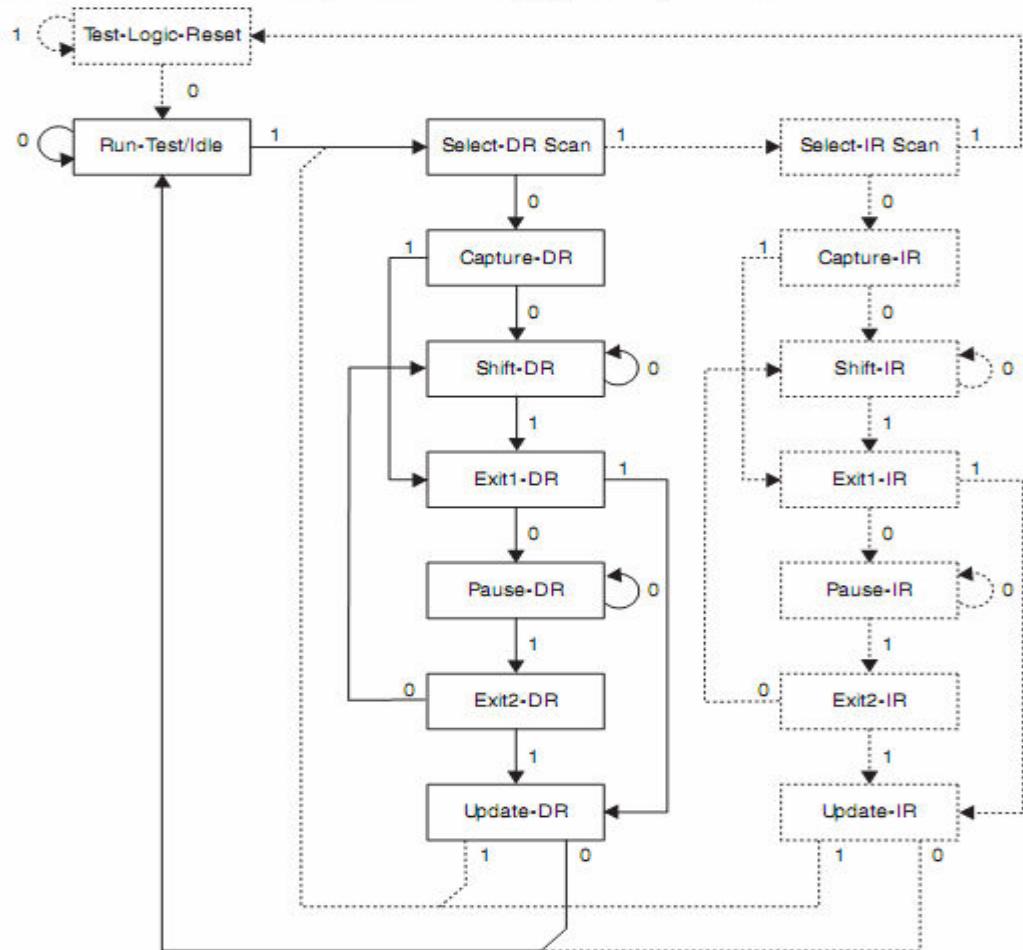
Table 130. JTAG Programming Instruction

Set **a** = address high bits, **b** = address low bits, **H** = 0 - Low byte, 1 - High Byte, **o** = data out, **i** = data in, **x** = don't care

Instruction	TDI sequence	TDO sequence	Notes
1a. Chip erase	0100011_10000000 0110001_10000000 0110011_10000000 0110011_10000000	xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx	
1b. Poll for chip erase complete	0110011_10000000	xxxxxxox_xxxxxxxx	(2)
2a. Enter Flash Write	0100011_00010000	xxxxxxxx_xxxxxxxx	
2b. Load Address High Byte	0000111_aaaaaaaaaa	xxxxxxxx_xxxxxxxx	(9)
2c. Load Address Low Byte	0000011_bbbaaaaa	xxxxxxxx_xxxxxxxx	
2d. Load Data Low Byte	0010011_iiiiiiii	xxxxxxxx_xxxxxxxx	
2e. Load Data High Byte	0010111_iiiiiiii	xxxxxxxx_xxxxxxxx	
2f. Latch Data	0110111_00000000 1110111_00000000 0110111_00000000	xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx	(1)
2g. Write Flash Page	0110111_00000000 0110101_00000000 0110111_00000000 0110111_00000000	xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx xxxxxxxx_xxxxxxxx	(1)
2h. Poll for Page Write complete	0110111_00000000	xxxxxxox_xxxxxxxx	(2)



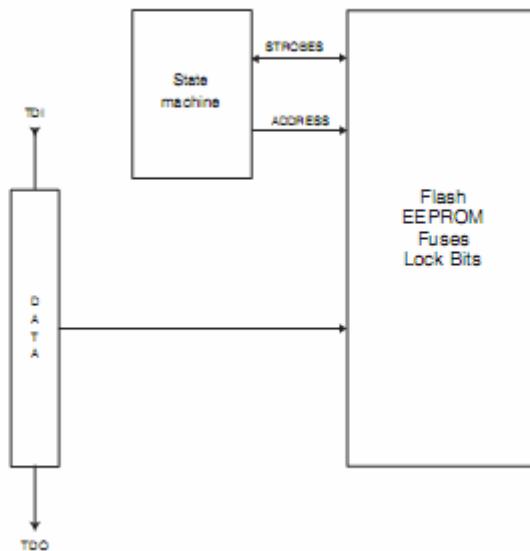
**Figure 149.** State Machine Sequence for Changing/Reading the Data Word



## Thanh ghi tải trang flash ảo.

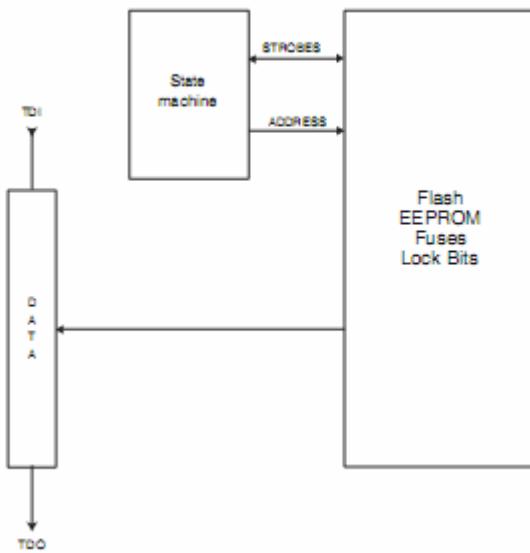
Thanh ghi tải trang flash ảo là một chuỗi quét ảo với độ dài bằng số lượng các bit trong một trang flash. Thanh ghi shift bên trong là 8 bit, và dữ liệu được chuyển một cách tự động đến bite bộ đệm trang flash bằng bite. Shift vào trong tất cả các từ lệnh trong một trang, việc bắt đầu các bit LSB của một lệnh đầu tiên trong một trang và kết thúc với bit MSB của lệnh cuối cùng trong trang. Điều này cung cấp một cách thuận tiện để tải trọng vẹn bộ đệm trang flash trước khi việc thực thi lệnh viết trang.

**Figure 150. Virtual Flash Page Load Register**



## Thanh ghi đọc trang flash ảo.

Thanh ghi đọc trang flash ảo là một chuỗi quét ảo với độ dài bằng số lượng các bit trong một trang flash. Thanh ghi shift bên trong là 8 bit, và dữ liệu được chuyển một cách tự động đến bite bộ đệm trang flash bằng bite. 8 chu kỳ đầu tiên được sử dụng để chuyển bite đầu tiên vào trong thanh ghi shift, và các bit mà được shift ra ngoài trong suốt 8 chu kỳ đó nên được bỏ qua. Tiếp theo qua trình khởi tạo này, dữ liệu được shift lại để bắt đầu bite LFB của lệnh đầu tiên trong trang và kết thúc với lệnh MSB của lệnh cuối cùng trong trang. Điều này cung cấp một cách thuận tiện để đọc một chương trình trong trang flash.

**Figure 151. Virtual Flash Page Read Register**

## Thuật toán lập trình

Tất cả các mẫu tham khảo bên dưới của loại “1a”, “1b”, tham khảo trang 130

### Sự truy nhập vào chế độ lập trình

1. Nhập lệnh JTAG AVR\_RESET và di chuyển 1 trong thanh ghi Reset .
2. Nhập lệnh PROG\_ENABLE và di chuyển 1010\_0011\_0111\_0000 trong thanh ghi kích hoạt lập trình

### Sự dời khỏi chế độ lập trình

1. nhập lệnh JTAG PROG\_COMMANDS
2. Vô hiệu hóa tất cả các lệnh lập trình bằng việc sử dụng lệnh không hoạt động 11a
3. nhập lệnh PROG\_ENABLE và di chuyển 0000\_0000\_0000\_0000 trong thanh ghi kích hoạt lập trình
4. nhập lệnh JTAG AVR\_RESET và di chuyển 0 trong thanh ghi Reset

### Tiến hành việc xóa chip

1. nhập lệnh JTAG PROG\_COMMANDS
2. bắt đầu xóa chip sử dụng lệnh lập trình 1a
3. kiểm tra vòng của việc xóa chip sử dụng lệnh lập trình 1b , hoặc đợi cho t<sub>VWLRH\_CE</sub> ( tham khảo bảng chú ý trên trang 299)

]

## Lập trình FLASH

Trước khi lập trình Flash 1 xóa chip (the Flash a chip erase ) phải được tiến hành . Xem “ việc tiến hành xóa “ trên 315

1. nhập lệnh JTAG PROG\_COMMANDS
  2. kích hoạt sự ghi Flash sử dụng lệnh lập trình 2a
  3. tải byte địa chỉ cao sử dụng lệnh lập trình 2b
  4. tải byte địa chỉ thấp sử dụng lệnh lập trình 2c
  5. tải dữ liệu sử dụng lệnh lập trình 2d , 2e , và 2f
  6. lặp lại bước 4 và 5 cho tất cả các từ lệnh trong trang
  7. viết trang sử dụng lệnh lập trình 2g
  8. hỏi vòng cho sự hoàn thành ghi Flash sử dụng lệnh lập trình 2h , hoặc đợi cho tWLRH (tham khảo bảng chú ý : trên trang 299)
  9. lặp lại bước 3 đến 7 cho đến khi tất cả các dữ liệu được lập trình
- Một sự chuyển phát dữ liệu hiệu quả hơn có thể được sử dụng để đạt được bằng việc sử dụng lệnh PROG\_PAGELOAD :
1. nhập lệnh JTAG PROG\_COMMANDS
  2. kích hoạt việc ghi Flash bằng việc sử dụng lệnh lập trình 2a
  3. tải địa chỉ trang sử dụng lệnh lập trình 2b và 2c . PCWORLD ( tham khảo bảng 123 trang 291) được sử dụng để đánh địa chỉ trong vòng 1 trang và phải được ghi là 0
  4. nhập lệnh JTAG PROG\_PAGELOAD
  5. tải toàn bộ trang bằng việc di chuyển trong tất cả các từ lệnh trong trang , bắt đầu với LSB của lệnh đầu tiên trong trang và kết thúc với MSB của lệnh cuối cùng trong trang
  6. nhập lệnh JTAG PROG\_COMMANDS
  7. viết trang sử dụng lệnh lập trình 2g
  8. hỏi vòng ghi Flash hoàn thành bằng việc sử dụng lệnh lập trình 2h , hoặc đợi tWLRH (tham khảo bảng chú ý trên trang 299)
  9. lặp lại từ bước 3 đến bước 8 cho đến khi tất cả dữ liệu được lập trình .

## Việc đọc dữ liệu Flash

1. Nhập lệnh JTAG PROG\_COMMANDS
  2. Kích hoạt đọc Flash sử dụng lệnh lập trình 3a
  3. tải địa chỉ sử dụng lệnh lập trình 3b và 3c
  4. đọc dữ liệu sử dụng lệnh lập trình 3d
  5. lặp lại từ bước 3 và 3 cho đến khi tất cả dữ liệu được đọc
- Một sự chuyển phát dữ liệu hiệu quả hơn có thể đạt được bằng việc sử dụng PROG\_PAGELOAD :
1. nhập lệnh JTAG PROG\_COMMANDS
  2. kích hoạt đọc Flash sử dụng lệnh lập trình 3a

3. tải địa chỉ trang bằng cách sử dụng lệnh lập trình 3b và 3c . PRWORD (tham khảo bảng 123 trang 291 ) được sử dụng đến địa chỉ trong vòng 1 trang và phải được ghi là 0
4. nhập lệnh JTAG PROG\_PAGEREAD
5. Đọc toàn bộ trang bằng việc di chuyển ra ngoài tất cả các từ lệnh trong trang , bắt đầu với LSB của lệnh đầu tiên trong trang và kết thúc với MSB của lệnh cuối cùng trong trang . Nhớ rằng 8 bit đầu tiên được di chuyển ra ngoài nên được bỏ qua .
6. nhập lệnh JTAG PROG\_COMMANDS
7. lặp lại bước 3 đến 6 cho đến khi tất cả dữ liệu được lập trình

## Lập trình EEPROM

Trước khi lập trình EEPROM 1 sự xóa chip phải được tiến hành . Xem phần “performing chip erase” trên trang 315

1. nhập lệnh JTAG PROG\_COMMANDS
2. kích hoạt viết EEPROM bằng việc sử dụng lệnh lập trình 4a
3. tải byte địa chỉ cao sử dụng lệnh lập trình 4b
4. tải byte địa chỉ thấp sử dụng lệnh lập trình 4c
5. tải dữ liệu sử dụng lệnh lập trình 4d và 4e
6. lặp lại bước 4 và 5 cho tất cả các byte dữ liệu trong trang
7. viết dữ liệu sử dụng lệnh lập trình 4f
8. hỏi vòng hoàn thành việc viết EEPROM sử dụng lệnh lập trình 4g , hoặc đợi twLRH (tham khảo bảng chú ý : trên trang 299)
9. lặp lại các bước từ 3 đến 8 cho đến khi tất cả dữ liệu được lập trình  
chú ý rằng lệnh PROG\_PAGELOAD không thể được sử dụng khi lập trình EEPROM

## Đọc EEPROM

1. Nhập lệnh JTAG PROG\_COMMANDS
  2. Kích hoạt đọc EEPROM sử dụng lệnh lập trình 5a
  3. Tải địa chỉ sử dụng các lệnh lập trình 5b và 5c
  4. Đọc dữ liệu sử dụng lệnh lập trình 5d
  5. Lặp lại bước 3 và 4 cho đến khi tất cả dữ liệu được đọc
- Chú ý rằng lệnh PROG\_PAGELOAD không thể được sử dụng khi đọc EEPROM

## Lập trình các cầu chì

1. Nhập lệnh JTAG PROG\_COMMANDS
2. Kích hoạt ghi Cầu chì sử dụng lệnh lập trình 6a

3. Tải byte dữ liệu sử dụng lệnh lập trình 6b . Một giá trị bit của “0” sẽ lập trình cầu chì tương ứng . Một giá trị “1” sẽ lập không lập trình cầu chì
4. Viết byte cầu chì mở rộng sử dụng lệnh lập trình 6c
5. Hỏi vòng cho việc ghi cầu chì hoàn thành sử dụng lệnh lập trình 6d , hoặc đợi t<sub>WRLH</sub> (tham khảo bảng chú ý trên trang 299)
6. Tải byte địa chỉ sử dụng lệnh lập trình 6e . Một giá trị bit của “0” sẽ lập trình cầu chì tương ứng , một giá trị “1” sẽ không lập trình cầu chì
7. Viết cầu chì byte cao sử dụng lệnh lập trình 6f
8. Hỏi vòng việc hoàn thành ghi cầu chì sử dụng lệnh lập trình 6g , hoặc đợi t<sub>WRLH</sub> (tham khảo bảng chú ý trên trang 299)
9. Tải byte dữ liệu sử dụng các lệnh lập trình 6h . Một giá trị “0” sẽ lập trình cầu chì , một giá trị “1” sẽ không lập trình cầu chì
10. Viết byte thấp cầu chì sử dụng lệnh lập trình 6i
11. Hỏi vòng hoàn thành việc ghi cầu chì sử dụng lệnh lập trình 6j , hoặc đợi t<sub>WRLH</sub> (tham khảo bảng chú ý trên trang 299)

## Lập trình các bit khóa

1. Nhập lệnh JTAG PROG\_COMMANDS
2. Kích hoạt ghi các bit khóa sử dụng lệnh lập trình 7a
3. Tải dữ liệu sử dụng lệnh lập trình 7b , Một giá trị của ‘0’ sẽ lập trình bit khóa tương ứng , một giá trị “1” sẽ di chuyển bit khóa không được nạp
4. Viết các bit khóa sử dụng lệnh lập trình 7c
5. Hỏi vòng cho việc hoàn thành ghi bit khóa sử dụng lệnh lập trình 7d , hoặc đợi t<sub>WRLH</sub> (tham khảo bảng chú ý trang 299)

## Đọc các bit cầu chì và các bit khóa

1. Nhập lệnh JTAG PROG\_COMMANDS
2. Kích hoạt đọc bit cầu chì/bit khóa sử dụng lệnh lập trình 8a
3. Để đọc tất cả các bit cầu chì và bit khóa sử dụng lệnh lập trình 8f
  - Chỉ đọc các bit cầu chì mở rộng sử dụng lệnh lập trình 8b
  - Chỉ đọc các byte cao cầu chì , sử dụng lệnh lập trình 8c
  - Chỉ đọc các byte cầu chì thấp sử dụng lệnh lập trình 8d
  - Chỉ đọc các bit khóa , sử dụng lệnh lập trình 8e

## Đọc các byte kí hiệu

1. Nhập lệnh JTAG PROGCOMMANS
2. Kích hoạt đọc các byte kí hiệu sử dụng lệnh lập trình 9a
3. Tải địa chỉ \$00 sử dụng lệnh lập trình 9b
4. Đọc byte kí hiệu đầu tiên sử dụng lệnh lập trình 9c

5. Lặp lại từ bước 3 và 4 với địa chỉ \$01 và \$02 để đọc byte kí hiệu thứ 2 và 3 một cách tương ứng

### **Đọc byte hiệu chỉnh**

1. Nhập lệnh JTAG PROG\_COMMANDS
2. Kích hoạt việc đọc byte hiệu chỉnh sử dụng lệnh lập trình 10a
3. Tải địa chỉ \$00 sử dụng lệnh lập trình 10b
4. Đọc byte hiệu chỉnh sử dụng lệnh lập trình 10c

## XXV . Các đặc tính điện

Chú ý : các giá trị thông thường chứa trong data sheet này được xây dựng trên cơ sở sự mô phỏng và các đặc tính của vi điều khiển AVR được chế tạo trên các quá trình công nghệ giống nhau . Các giá trị Min và Max sẽ có thể sử dụng sau khi thiết bị được tiêu chuẩn hóa

### Chỉ số cực đại tuyệt đối

#### Absolute Maximum Ratings\*

Operating Temperature .....	-55°C to +125°C
Storage Temperature .....	-65°C to +150°C
Voltage on any Pin except RESET with respect to Ground .....	-0.5V to V <sub>CC</sub> +0.5V
Voltage on RESET with respect to Ground.....	-0.5V to +13.0V
Maximum Operating Voltage .....	6.0V
DC Current per I/O Pin .....	40.0 mA
DC Current V <sub>CC</sub> and GND Pins .....	200.0 - 400.0mA

Thông báo : các giá trị ngoài giới hạn được liệt kê trong Bảng chỉ số cực đại tuyệt đối bên dưới có thể gây ra hỏng hóc vĩnh viễn cho thiết bị . Đây chỉ là một giá trị giới hạn và chức năng điều khiển của thiết bị tại các điều kiện này hoặc các điều kiện khác bên ngoài các thông số sau thì không được sử dụng . Sự đưa ra của các điều kiện chỉ số cực đại tuyệt đối cho chu kỳ mở rộng có thể ảnh hưởng đến các thiết bị liên quan .

$T_A = -40^\circ\text{C}$  to  $85^\circ\text{C}$ ,  $V_{CC} = 2.7\text{V}$  to  $5.5\text{V}$  (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
$V_{IL}$	Input Low Voltage except XTAL1 and <u>RESET</u> pins	$V_{CC}=2.7$ - $5.5$ .	-0.5		$0.2 V_{CC}^{(1)}$	V
$V_{IH}$	Input High Voltage except XTAL1 and <u>RESET</u> pins	$V_{CC}=2.7$ - $5.5$ .	$0.6 V_{CC}^{(2)}$		$V_{CC} + 0.5$	V
$V_{IL1}$	Input Low Voltage XTAL1 pin	$V_{CC}=2.7$ - $5.5$ .	-0.5		$0.1 V_{CC}^{(1)}$	V
$V_{IH1}$	Input High Voltage XTAL1 pin	$V_{CC}=2.7$ - $5.5$ .	$0.7 V_{CC}^{(2)}$		$V_{CC} + 0.5$	V
$V_{IL2}$	Input Low Voltage <u>RESET</u> pin	$V_{CC}=2.7$ - $5.5$ .	-0.5		$0.2 V_{CC}^{(1)}$	V
$V_{IH2}$	Input High Voltage <u>RESET</u> pin	$V_{CC}=2.7$ - $5.5$ .	$0.85 V_{CC}^{(2)}$		$V_{CC} + 0.5$	V
$V_{OL}$	Output Low Voltage <sup>(3)</sup> (Ports A,B,C,D, E, F, G)	$I_{OL} = 20 \text{ mA}, V_{CC} = 5\text{V}$ $I_{OL} = 10 \text{ mA}, V_{CC} = 3\text{V}$			0.7 0.5	V
$V_{OH}$	Output High Voltage <sup>(4)</sup> (Ports A,B,C,D, E, F, G)	$I_{OH} = -20 \text{ mA}, V_{CC} = 5\text{V}$ $I_{OH} = -10 \text{ mA}, V_{CC} = 3\text{V}$	4.2 2.2			V
$I_{IL}$	Input Leakage Current I/O Pin	$V_{CC} = 5.5\text{V}$ , pin low (absolute value)			1.0	$\mu\text{A}$
$I_{IH}$	Input Leakage Current I/O Pin	$V_{CC} = 5.5\text{V}$ , pin high (absolute value)			1.0	$\mu\text{A}$
$R_{RST}$	Reset Pull-up Resistor		30		60	$\text{k}\Omega$
$R_{PEN}$	PEN Pull-up Resistor		30		60	$\text{k}\Omega$
$R_{PU}$	I/O Pin Pull-up Resistor		20		50	$\text{k}\Omega$

$T_A = -40^\circ\text{C}$  to  $85^\circ\text{C}$ ,  $V_{CC} = 2.7\text{V}$  to  $5.5\text{V}$  (unless otherwise noted) (Continued)

Symbol	Parameter	Condition	Min	Typ	Max	Units
$I_{CC}$	Power Supply Current	Active 4 MHz, $V_{CC} = 3\text{V}$ (ATmega128L)		5	5.5	$\text{mA}$
		Active 8 MHz, $V_{CC} = 5\text{V}$ (ATmega128)		17	19	$\text{mA}$
		Idle 4 MHz, $V_{CC} = 3\text{V}$ (ATmega128L)		2	2.5	$\text{mA}$
		Idle 8 MHz, $V_{CC} = 5\text{V}$ (ATmega128)		8	11	$\text{mA}$
	Power-down mode	WDT enabled, $V_{CC} = 3\text{V}$		< 15	25	$\mu\text{A}$
		WDT disabled, $V_{CC} = 3\text{V}$		< 5	10	$\mu\text{A}$
$V_{ACIO}$	Analog Comparator Input Offset Voltage	$V_{CC} = 5\text{V}$ $V_{in} = V_{CC}/2$			40	$\text{mV}$
$I_{ACLK}$	Analog Comparator Input Leakage Current	$V_{CC} = 5\text{V}$ $V_{in} = V_{CC}/2$	-50		50	$\text{nA}$
$t_{ACPD}$	Analog Comparator Propagation Delay	$V_{CC} = 2.7\text{V}$ $V_{CC} = 5.0\text{V}$		750 500		ns

- Notes:
1. "Max" means the highest value where the pin is guaranteed to be read as low
  2. "Min" means the lowest value where the pin is guaranteed to be read as high
  3. Although each I/O port can sink more than the test conditions (20 mA at  $V_{CC} = 5\text{V}$ , 10 mA at  $V_{CC} = 3\text{V}$ ) under steady state conditions (non-transient), the following must be observed:

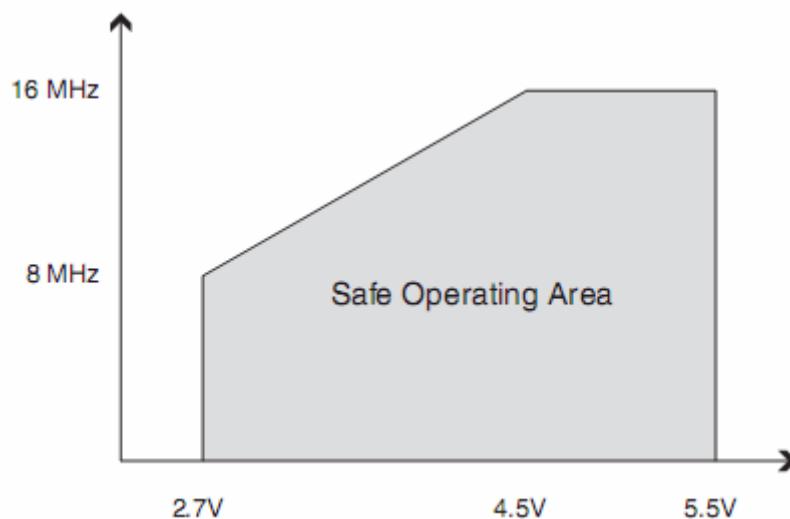
Chú ý :

1. "Max" có nghĩa là giá trị cao nhất ở đó chân được đảm bảo được đọc như là mức thấp
2. "Min" nghĩa là giá trị thấp nhất mà ở đó chân được đảm bảo được đọc như là mức cao
3. Mặc dù mỗi cổng I/O có thể tản nhiệt nhiều hơn điều kiện kiểm định ( 20 mA at VCC = 5V, 10 mA at VCC = 3V) dưới các điều kiện trạng thái ổn định ( không có quá độ ) , các điều bên dưới đây phải được quan sát :
  - 1] The sum of all IOL, for all ports, should not exceed 400 mA.
  - 2] The sum of all IOL, for ports A0 - A7, G2, C3 - C7 should not exceed 100 mA.
  - 3] The sum of all IOL, for ports C0 - C2, G0 - G1, D0 - D7, XTAL2 should not exceed 100 mA.
  - 4] The sum of all IOL, for ports B0 - B7, G3 - G4, E0 - E7 should not exceed 100 mA.
  - 5] The sum of all IOL, for ports F0 - F7, should not exceed 100 mA.  
If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
4. Mặc dù mỗi cổng I/O có thể tản nhiệt nhiều hơn điều kiện kiểm định ( 20 mA at VCC = 5V, 10 mA at VCC = 3V) dưới các điều kiện trạng thái ổn định ( không có quá độ ) , các điều bên dưới đây phải được quan sát :
  - 1] The sum of all IOH, for all ports, should not exceed 400 mA.
  - 2] The sum of all IOH, for ports A0 - A7, G2, C3 - C7 should not exceed 100 mA.
  - 3] The sum of all IOH, for ports C0 - C2, G0 - G1, D0 - D7, XTAL2 should not exceed 100 mA.
  - 4] The sum of all IOH, for ports B0 - B7, G3 - G4, E0 - E7 should not exceed 100 mA.
  - 5] The sum of all IOH, for ports F0 - F7, should not exceed 100 mA.  
If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.

]

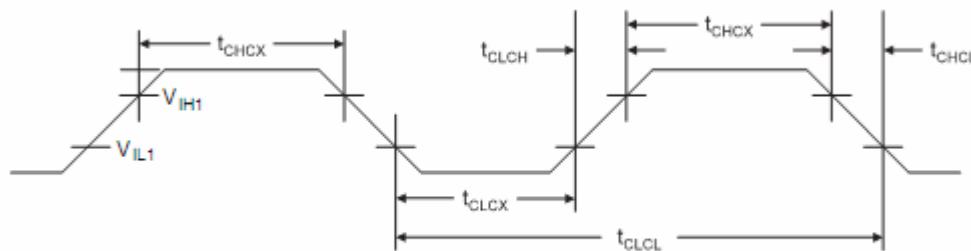
## Độ dốc tốc độ

Hình 152 : Tần số cực đại và V<sub>CC</sub>



## Dạng xung điều khiển xung nhịp bên ngoài

Figure 153. External Clock Drive Waveforms



## Điều khiển xung nhịp bên ngoài

Table 131. External Clock Drive

Symbol	Parameter	V <sub>CC</sub> = 2.7V to 5.5V		V <sub>CC</sub> = 4.5V to 5.5V		Units
		Min	Max	Min	Max	
1/t <sub>OLCL</sub>	Oscillator Frequency	0	8	0	16	MHz
t <sub>CLCL</sub>	Clock Period	125		62.5		ns
t <sub>CHCX</sub>	High Time	50		25		ns
t <sub>CLCX</sub>	Low Time	50		25		ns
t <sub>CLCH</sub>	Rise Time			1.6		μs
t <sub>CHCL</sub>	Fall Time			1.6		μs
Δt <sub>CLCL</sub>	Change in period from one clock cycle to the next		2		2	%

**Table 132.** External RC Oscillator, Typical Frequencies

R [kΩ] <sup>(1)</sup>	C [pF]	f <sup>(2)</sup>
33	22	650 kHz
10	22	2.0 MHz

Notes: 1. R should be in the range 3 kΩ - 100 kΩ, and C should be at least 20 pF. The C values given in the table includes pin capacitance. This will vary with package type.

2. The frequency will vary with package type and board layout.

Bảng 132 : bộ tạo dao động RC bên ngoài , tần số thông thường

### Thông số kĩ thuật của giao diện 2 dây nối tiếp

Bảng 133 miêu tả các thiết bị cần thiết để kết nối với bus 2 dây nối tiếp . Giao diện 2 dây nối tiếp của Atmega 128 bằng hoặc vượt quá các yêu cầu này dưới các điều kiện đã được chú ý .

Các kí hiệu thời gian được tham chiếu đến bảng 154

**Table 133.** Two-wire Serial Bus Requirements

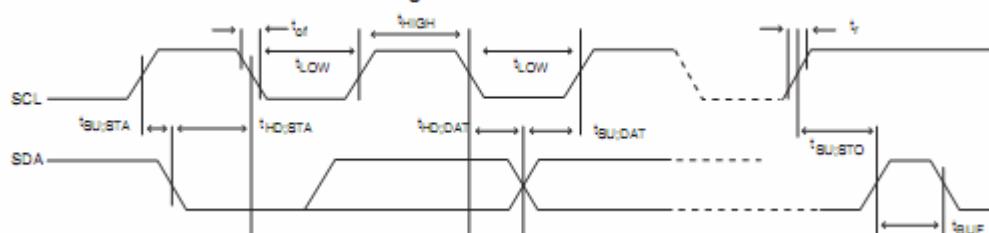
Symbol	Parameter	Condition	Min	Max	Units
V <sub>L</sub>	Input Low-voltage		-0.5	0.3 V <sub>CC</sub>	V
V <sub>H</sub>	Input High-voltage		0.7 V <sub>CC</sub>	V <sub>CC</sub> + 0.5	V
V <sub>HYS</sub> <sup>(1)</sup>	Hysteresis of Schmitt Trigger Inputs		0.05 V <sub>CC</sub> <sup>(2)</sup>	—	V
V <sub>OLO</sub> <sup>(1)</sup>	Output Low-voltage	3 mA sink current	0	0.4	V
t <sub>r</sub> <sup>(1)</sup>	Rise Time for both SDA and SCL		20 + 0.1C <sub>b</sub> <sup>(3)(4)</sup>	300	ns
t <sub>f</sub> <sup>(1)</sup>	Output Fall Time from V <sub>HMIN</sub> to V <sub>LMAX</sub>	10 pF < C <sub>b</sub> < 400 pF <sup>(3)</sup>	20 + 0.1C <sub>b</sub> <sup>(3)(4)</sup>	250	ns
t <sub>sp</sub> <sup>(1)</sup>	Spikes Suppressed by Input Filter		0	50 <sup>(5)</sup>	ns
I <sub>I</sub>	Input Current each I/O Pin	0.1 V <sub>CC</sub> < V <sub>I</sub> < 0.9 V <sub>CC</sub>	-10	10	µA
C <sub>I</sub> <sup>(6)</sup>	Capacitance for each I/O Pin		—	10	pF
f <sub>SCL</sub>	SCL Clock Frequency	f <sub>SCL</sub> <sup>(4)</sup> > max(16f <sub>SCL</sub> , 250kHz) <sup>(5)</sup>	0	400	kHz
R <sub>P</sub>	Value of Pull-up resistor	f <sub>SCL</sub> ≤ 100 kHz	$\frac{V_{CC} - 0.4V}{3mA}$	$\frac{1000ns}{C_b}$	Ω
		f <sub>SCL</sub> > 100 kHz	$\frac{V_{CC} - 0.4V}{3mA}$	$\frac{300ns}{C_b}$	Ω
t <sub>holdX</sub>	Hold Time (repeated) START Condition	f <sub>SCL</sub> ≤ 100 kHz	4.0	—	µs
		f <sub>SCL</sub> > 100 kHz	0.6	—	µs
t <sub>low</sub>	Low Period of the SCL Clock	f <sub>SCL</sub> ≤ 100 kHz <sup>(6)</sup>	4.7	—	µs
		f <sub>SCL</sub> > 100 kHz <sup>(7)</sup>	1.3	—	µs
t <sub>high</sub>	High period of the SCL clock	f <sub>SCL</sub> ≤ 100 kHz	4.0	—	µs
		f <sub>SCL</sub> > 100 kHz	0.6	—	µs
t <sub>setupX</sub>	Set-up time for a repeated START condition	f <sub>SCL</sub> ≤ 100 kHz	4.7	—	µs
		f <sub>SCL</sub> > 100 kHz	0.6	—	µs
t <sub>holdX</sub>	Data hold time	f <sub>SCL</sub> ≤ 100 kHz	0	3.45	µs
		f <sub>SCL</sub> > 100 kHz	0	0.9	µs
t <sub>setupX</sub>	Data setup time	f <sub>SCL</sub> ≤ 100 kHz	250	—	ns
		f <sub>SCL</sub> > 100 kHz	100	—	ns
t <sub>setupX</sub>	Setup time for STOP condition	f <sub>SCL</sub> ≤ 100 kHz	4.0	—	µs
		f <sub>SCL</sub> > 100 kHz	0.6	—	µs
t <sub>burst</sub>	Bus free time between a STOP and START condition	f <sub>SCL</sub> ≤ 100 kHz	4.7	—	µs

Chú ý :

1. trong Atmega 128 , tham số này được chuẩn hóa và không được kiểm tra 100%
2. chỉ cần thiết cho f<sub>SCL</sub> > 100 kHz
3. C<sub>b</sub> = điện dung của một đường line trong pF

4. các yêu cầu này phải được áp dụng cho tất cả các quá trình điều khiển giao diện 2 dây nối tiếp cho Atmega 128
  5.  $f_{CK}$  = tần số xung nhịp CPU
  6. chu kì thấp thật sự sinh ra bởi giao diện 2 dây nối tiếp Atmega 128 thì  $(1/f_{SCL} - 2/f_{CK})$  dù cho  $f_{CK}$  phải lớn hơn 6 MHz cho thời gian cần thiết để đạt đến  $f_{SCL} = 100\text{kHz}$
  7. các chu kì thấp thật sự được sinh ra bởi giao diện nối tiếp của Atmega 128 là  $(1/f_{SCL} - 2/f_{CK})$  dù cho thời gian thấp cần thiết sẽ không được gấp chính xác cho  $f_{SCL} > 308\text{kHz}$ . các thiết bị Atmega 128 được kết nối đến bus có thể giao tiếp ở tất cả các tốc độ (400kHz) với các thiết bị Atmega 128 khác , tốt như bất cứ các thiết bị khác với bản thân  $t_{low}$

**Figure 154.** Two-wire Serial Bus Timing



## Các đặc tính của bộ định thời SPI

Table 134. SPI Timing Parameters

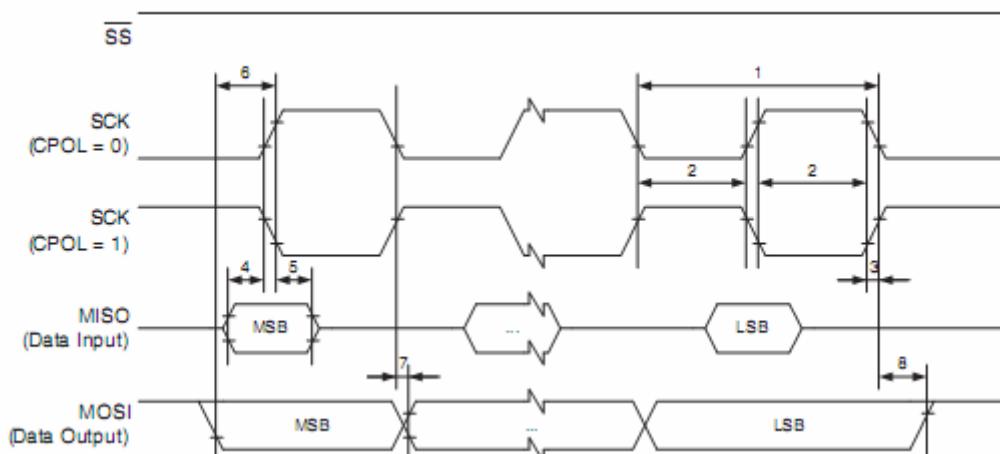
	Description	Mode	Min	Typ	Max	
1	SCK period	Master		See Table 72		ns
2	SCK high/low	Master		50% duty cycle		
3	Rise/Fall time	Master		3.6		
4	Setup	Master		10		
5	Hold	Master		10		
6	Out to SCK	Master		$0.5 \cdot t_{sck}$		
7	SCK to out	Master		10		
8	SCK to out high	Master		10		
9	SS low to out	Slave		15		
10	SCK period	Slave	$4 \cdot t_{ck}$			
11	SCK high/low <sup>(1)</sup>	Slave	$2 \cdot t_{ck}$			
12	Rise/Fall time	Slave			1.6	μs
13	Setup	Slave	10			
14	Hold	Slave	10			
15	SCK to out	Slave		15		
16	SCK to SS high	Slave	20			
17	SS high to tri-state	Slave		10		
18	SS low to SCK	Slave	$2 \cdot t_{ck}$			

Note: 1. In SPI Programming mode the minimum SCK high/low period is:

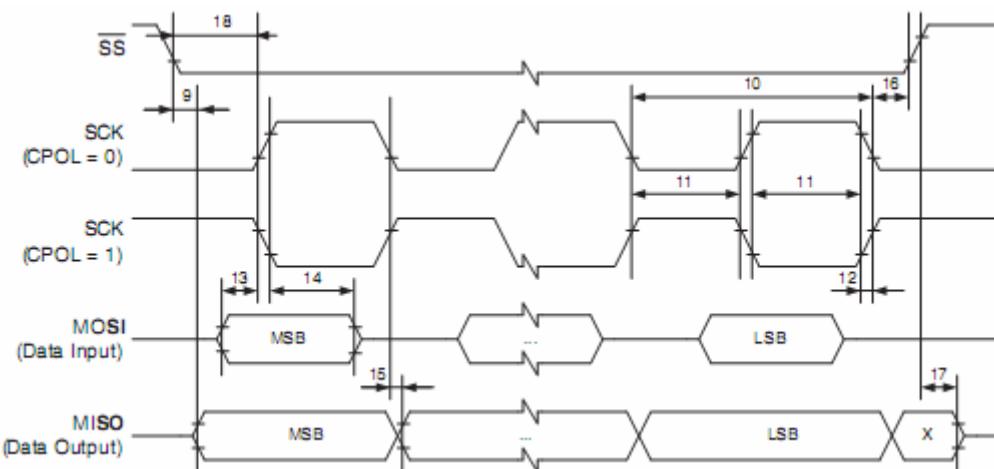
- $2 \cdot t_{CKL}$  for  $f_{CK} < 12$  MHz
- $3 \cdot t_{CKH}$  for  $f_{CK} > 12$  MHz

Hình 155 : các điều kiện cần của giao diện SPI (chế độ Master )

Figure 155. SPI Interface Timing Requirements (Master Mode)



Hình 156 : các điều kiện cần của giao diện SPI (chế độ Slave )



## Các chỉ số ADC

Bảng 135 : các chỉ số ADC , các kênh single ended

Table 135. ADC Characteristics, Single Ended Channels

Symbol	Parameter	Condition	MIn <sup>(1)</sup>	Typ <sup>(1)</sup>	Max <sup>(1)</sup>	Units
	Resolution	Single Ended Conversion			10	Bits
	Absolute Accuracy (Including INL, DNL, Quantization Error, Gain and Offset Error)	Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 200 kHz		1.5		LSB
		Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 1 MHz		3.25		LSB
		Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 200 kHz Noise Reduction mode		1.5		LSB
		Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 1 MHz Noise Reduction mode		3.75		LSB
	Integral Non-Linearity (INL)	Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 200 kHz		0.75		LSB
	Differential Non-Linearity (DNL)	Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 200 kHz		0.5		LSB
	Gain Error	Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 200 kHz		1		LSB
	Offset error	Single Ended Conversion $V_{REF} = 4V$ , $V_{CC} = 4V$ ADC clock = 200 kHz		1		LSB
	Clock Frequency		50		1000	kHz
	Conversion Time		13		260	μs
AVCC	Analog Supply Voltage		$V_{CC} - 0.3^{(2)}$		$V_{CC} + 0.3^{(3)}$	V
$V_{REF}$	Reference Voltage		2.0		AVCC	V
$V_N$	Input Voltage		GND		$V_{REF}$	V
	Input Bandwidth				38.5	kHz
$V_{INT}$	Internal Voltage Reference		2.3	2.56	2.7	V
$R_{REF}$	Reference Input Resistance			32		kΩ
$R_{AIN}$	Analog Input Resistance		55	100		MΩ

Notes: 1. Values are guidelines only.

2. Minimum for AVCC is 2.7V.

3. Maximum for AVCC is 5.5V

Bảng 136 : các chỉ tiêu ADC , các kênh riêng biệt

Symbol	Parameter	Condition	Min <sup>(1)</sup>	Typ <sup>(1)</sup>	Max <sup>(1)</sup>	Units
	Resolution	Gain = 1x			10	Bits
		Gain = 10x			10	Bits
		Gain = 200x			10	Bits
	Absolute Accuracy	Gain = 1x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		17		LSB
		Gain = 10x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		17		LSB
		Gain = 200x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		7		LSB
	Integral Non-Linearity (INL) (Accuracy after Calibration for Offset and Gain Error)	Gain = 1x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		1.5		LSB
		Gain = 10x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		2		LSB
		Gain = 200x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		5		LSB
	Gain Error	Gain = 1x		1.5		%
		Gain = 10x		1.5		%
		Gain = 200x		0.5		%
	Offset Error	Gain = 1x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		2		LSB
		Gain = 10x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		3		LSB
		Gain = 200x $V_{REF} = 4V, V_{CC} = 5V$ ADC clock = 50 - 200 kHz		4		LSB
	Clock Frequency		50		200	kHz
	Conversion Time		65		260	μs
AVCC	Analog Supply Voltage		$V_{CC} - 0.3^{(2)}$		$V_{CC} + 0.3^{(3)}$	V
$V_{REF}$	Reference Voltage		2.0		AVCC - 0.5	V
$V_N$	Input Voltage		GND		$V_{CC}$	V
$V_{DIFF}$	Input Differential Voltage		$-V_{REF}/\text{Gain}$		$V_{REF}/\text{Gain}$	V
	ADC Conversion Output		-511		511	LSB
	Input Bandwidth			4		kHz
Symbol	Parameter	Condition	Min <sup>(1)</sup>	Typ <sup>(1)</sup>	Max <sup>(1)</sup>	Units
$V_{INT}$	Internal Voltage Reference		2.3	2.56	2.7	V
$R_{REF}$	Reference Input Resistance			32		kΩ
$R_{AIN}$	Analog Input Resistance		55	100		MΩ

Notes:

1. Values are guidelines only.
2. Minimum for AVCC is 2.7V.
3. Maximum for AVCC is 5.5V.

## Sự định thời bộ nhớ dữ liệu bên ngoài

Bảng 137: các chỉ số bộ nhớ dữ liệu bên ngoài , 4,5 đến 5,5 V , không có trạng thái chờ

Symbol	Parameter	8 MHz Oscillator		Variable Oscillator		Unit
		Min	Max	Min	Max	
0	$t_{f_{CLOCK}}$	Oscillator Frequency		0.0	16	MHz
1	$t_{LHL}$	ALE Pulse Width	115	$1.0t_{CLOCK}\cdot10$		ns
2	$t_{AVLL}$	Address Valid A to ALE Low	57.5	$0.5t_{CLOCK}\cdot5^{(1)}$		ns
3a	$t_{LAX_ST}$	Address Hold After ALE Low, write access	5	5		ns
3b	$t_{LAX_LD}$	Address Hold after ALE Low, read access	5	5		ns
4	$t_{AVLC}$	Address Valid C to ALE Low	57.5	$0.5t_{CLOCK}\cdot5^{(1)}$		ns
5	$t_{AVRL}$	Address Valid to RD Low	115	$1.0t_{CLOCK}\cdot10$		ns
6	$t_{AWL}$	Address Valid to WR Low	115	$1.0t_{CLOCK}\cdot10$		ns
7	$t_{LWL}$	ALE Low to WR Low	47.5	67.5	$0.5t_{CLOCK}\cdot15^{(2)}$	$0.5t_{CLOCK}\cdot5^{(2)}$
8	$t_{URL}$	ALE Low to RD Low	47.5	67.5	$0.5t_{CLOCK}\cdot15^{(2)}$	$0.5t_{CLOCK}\cdot5^{(2)}$
9	$t_{DVRH}$	Data Setup to RD High	40	40		ns
10	$t_{RLDV}$	Read Low to Data Valid		75	$1.0t_{CLOCK}\cdot50$	ns
11	$t_{PHDX}$	Data Hold After RD High	0	0		ns
12	$t_{RLRH}$	RD Pulse Width	115	$1.0t_{CLOCK}\cdot10$		ns
13	$t_{DWL}$	Data Setup to WR Low	42.5	$0.5t_{CLOCK}\cdot20^{(1)}$		ns
14	$t_{WHDX}$	Data Hold After WR High	115	$1.0t_{CLOCK}\cdot10$		ns
15	$t_{DWHH}$	Data Valid to WR High	125	$1.0t_{CLOCK}$		ns
16	$t_{WLWH}$	WR Pulse Width	115	$1.0t_{CLOCK}\cdot10$		ns

Note: 1. This assumes 50% clock duty cycle. The half period is actually the high time of the external clock, XTAL1.

2. This assumes 50% clock duty cycle. The half period is actually the low time of the external clock, XTAL1.

Bảng 138 các chỉ tiêu bộ nhớ dữ liệu bên ngoài , 4,5 -5,5 V , 1 chu kì trạng thái chờ

Symbol	Parameter	8 MHz Oscillator		Variable Oscillator		Unit
		Min	Max	Min	Max	
0	$t_{f_{CLOCK}}$	Oscillator Frequency		0.0	16	MHz
10	$t_{RLDV}$	Read Low to Data Valid		200	$2.0t_{CLOCK}\cdot50$	ns
12	$t_{RLRH}$	RD Pulse Width	240	$2.0t_{CLOCK}\cdot10$		ns
15	$t_{DWHH}$	Data Valid to WR High	240	$2.0t_{CLOCK}$		ns
16	$t_{WLWH}$	WR Pulse Width	240	$2.0t_{CLOCK}\cdot10$		ns

Bảng 139 : chỉ số bộ nhớ dữ liệu bên ngoài , 4,5-5,5 V , SRWn1 =1 , SRWn0 =0

Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
		Min	Max	Min	Max	
0	$t_{f_{CLOCK}}$	Oscillator Frequency		0.0	16	MHz
10	$t_{RLDV}$	Read Low to Data Valid		325	$3.0t_{CLOCK}\cdot50$	ns
12	$t_{RLRH}$	RD Pulse Width	365	$3.0t_{CLOCK}\cdot10$		ns
15	$t_{DWHH}$	Data Valid to WR High	375	$3.0t_{CLOCK}$		ns
16	$t_{WLWH}$	WR Pulse Width	365	$3.0t_{CLOCK}\cdot10$		ns

Bảng 140 : các chỉ số bộ nhớ dữ liệu bên ngoài 4,5 – 5,5V , SRWn1 =1 , SRWn0 =1

	Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
			Min	Max	Min	Max	
0	$t_{tQCL}$	Oscillator Frequency			0.0	18	MHz
10	$t_{RLDV}$	Read Low to Data Valid		325		$3.0t_{tQCL}-50$	ns
12	$t_{RLRH}$	RD Pulse Width	365		$3.0t_{tQCL}-10$		ns
14	$t_{WHDX}$	Data Hold After WR High	240		$2.0t_{tQCL}-10$		ns
15	$t_{DWWH}$	Data Valid to WR High	375		$3.0t_{tQCL}$		ns
16	$t_{WLWH}$	WR Pulse Width	365		$3.0t_{tQCL}-10$		ns

Bảng 141 : chỉ số bộ nhớ dữ liệu ngoài , 2,7 – 5,5 V , không có trạng thái chờ

	Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
			Min	Max	Min	Max	
0	$t_{tQCL}$	Oscillator Frequency			0.0	8	MHz
1	$t_{tHLL}$	ALE Pulse Width	235		$t_{tQCL}-15$		ns
2	$t_{tAVLL}$	Address Valid A to ALE Low	115		$0.5t_{tQCL}-10^{(1)}$		ns
3a	$t_{tLAX_ST}$	Address Hold After ALE Low, write access	5		5		ns
3b	$t_{tLAX_LD}$	Address Hold after ALE Low, read access	5		5		ns
4	$t_{tAVLLO}$	Address Valid C to ALE Low	115		$0.5t_{tQCL}-10^{(1)}$		ns
5	$t_{tAVRL}$	Address Valid to RD Low	235		$1.0t_{tQCL}-15$		ns
6	$t_{tAVWL}$	Address Valid to WR Low	235		$1.0t_{tQCL}-15$		ns
7	$t_{tLWL}$	ALE Low to WR Low	115	130	$0.5t_{tQCL}-10^{(2)}$	$0.5t_{tQCL}+5^{(2)}$	ns
8	$t_{tRLL}$	ALE Low to RD Low	115	130	$0.5t_{tQCL}-10^{(2)}$	$0.5t_{tQCL}+5^{(2)}$	ns
9	$t_{tDVRH}$	Data Setup to RD High	45		45		ns
10	$t_{tRLDV}$	Read Low to Data Valid		190		$1.0t_{tQCL}-60$	ns
11	$t_{tRHDX}$	Data Hold After RD High	0		0		ns

	Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
			Min	Max	Min	Max	
12	$t_{tRLRH}$	RD Pulse Width	235		$1.0t_{tQCL}-15$		ns
13	$t_{tDWL}$	Data Setup to WR Low	105		$0.5t_{tQCL}-20^{(1)}$		ns
14	$t_{tWHDX}$	Data Hold After WR High	235		$1.0t_{tQCL}-15$		ns
15	$t_{tDWWH}$	Data Valid to WR High	250		$1.0t_{tQCL}$		ns
16	$t_{tWLWH}$	WR Pulse Width	235		$1.0t_{tQCL}-15$		ns

Notes: 1. This assumes 50% clock duty cycle. The half period is actually the high time of the external clock, XTAL1.

2. This assumes 50% clock duty cycle. The half period is actually the low time of the external clock, XTAL1.

Bảng 142 : các chỉ số bộ nhớ dữ liệu bên ngoài , 2,7 – 5,5 V SRWn1 =1 , SRWn0 =1

	Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
			Min	Max	Min	Max	
0	$t_{tQCL}$	Oscillator Frequency			0.0	8	MHz
10	$t_{tRLDV}$	Read Low to Data Valid		440		$2.0t_{tQCL}-60$	ns
12	$t_{tRLRH}$	RD Pulse Width	485		$2.0t_{tQCL}-15$		ns
15	$t_{tDWWH}$	Data Valid to WR High	500		$2.0t_{tQCL}$		ns
16	$t_{tWLWH}$	WR Pulse Width	485		$2.0t_{tQCL}-15$		ns

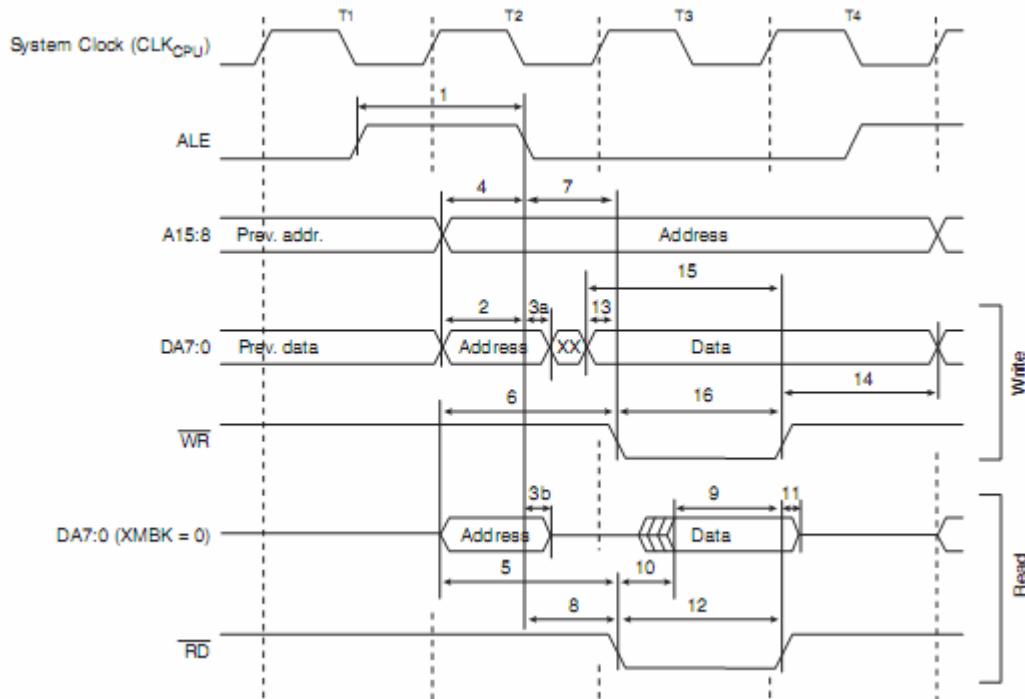
Bảng 143 : các chỉ số bộ nhớ dữ liệu bên ngoài , 2,7 – 5,5 V SRWn1 =1 , SRWn0 =0

	Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
			Min	Max	Min	Max	
0	$t_{tQCL}$	Oscillator Frequency			0.0	8	MHz
10	$t_{tRLDV}$	Read Low to Data Valid		690		$3.0t_{tQCL}-60$	ns
12	$t_{tRLRH}$	RD Pulse Width	735		$3.0t_{tQCL}-15$		ns
15	$t_{tDWWH}$	Data Valid to WR High	750		$3.0t_{tQCL}$		ns
16	$t_{tWLWH}$	WR Pulse Width	735		$3.0t_{tQCL}-15$		ns

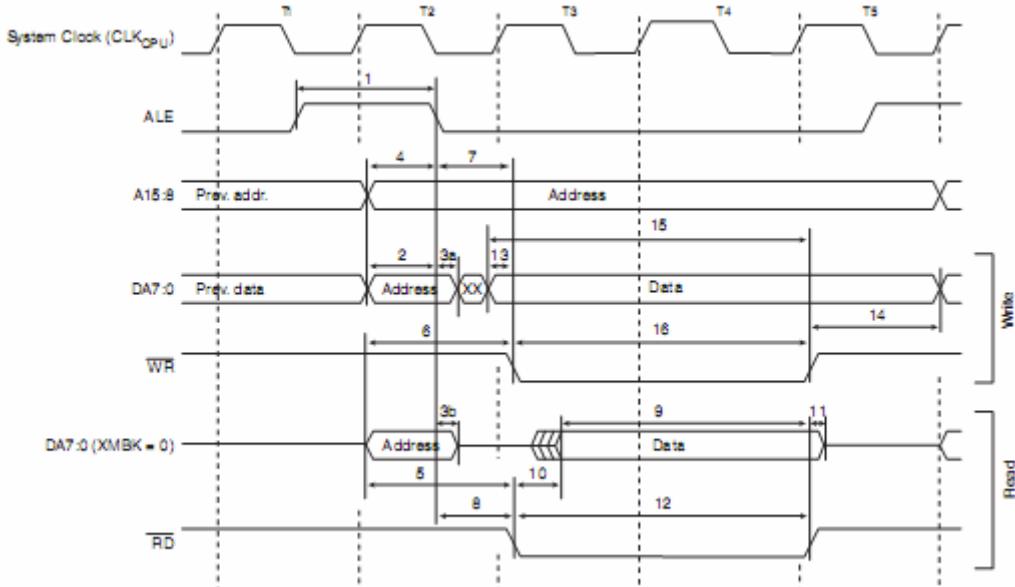
Bảng 144 : các chỉ số bộ nhớ dữ liệu bên ngoài ,2,7 – 5,5 V SRWn1 =1 , SRWn0 =0

	Symbol	Parameter	4 MHz Oscillator		Variable Oscillator		Unit
			Min	Max	Min	Max	
0	$t_{f_{OCL}}$	Oscillator Frequency			0.0	8	MHz
10	$t_{RLDV}$	Read Low to Data Valid		690		$3.0t_{f_{OCL}}-60$	ns
12	$t_{RLRH}$	RD Pulse Width	735		$3.0t_{f_{OCL}}-15$		ns
14	$t_{WHDX}$	Data Hold After WR High	485		$2.0t_{f_{OCL}}-15$		ns
15	$t_{WVH}$	Data Valid to WR High	750		$3.0t_{f_{OCL}}$		ns
16	$t_{WLWH}$	WR Pulse Width	735		$3.0t_{f_{OCL}}-15$		ns

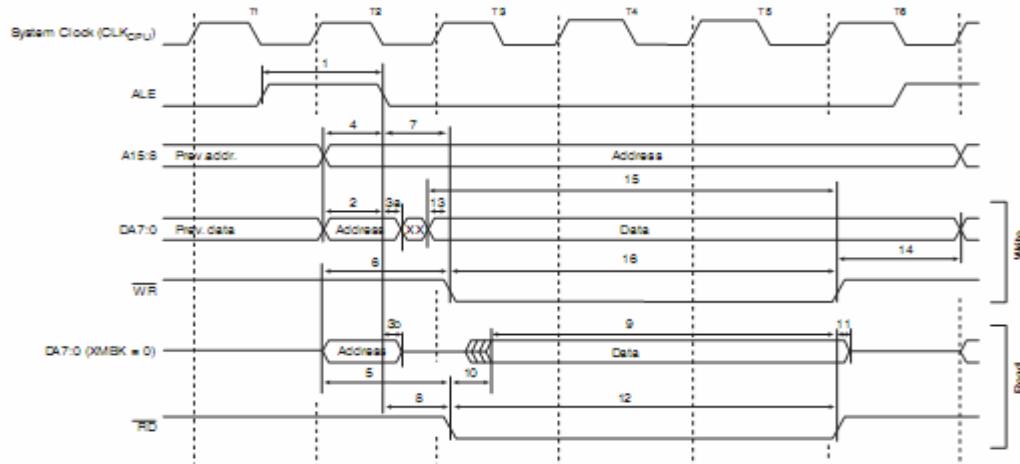
Hình 157 : giản đồ thời gian bộ nhớ dữ liệu bên ngoài (SRWn1 = 0 , SRWn0 = 0 )



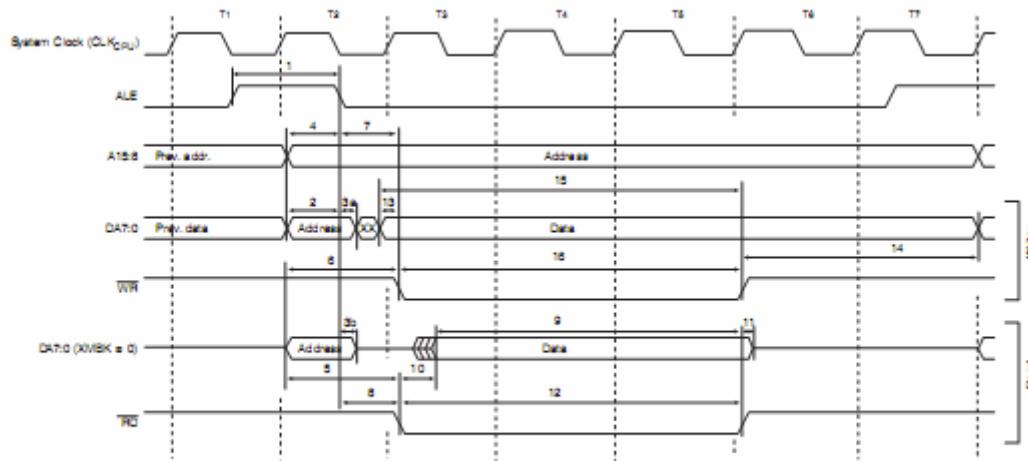
Hình 158 : giản đồ thời gian của bộ nhớ bên ngoài (SRWn1 = 0 , SRWn0 = 0 )



Hình 159 : giản đồ thời gian bộ nhớ bên ngoài (SRWn1 = 0 , SRWn0 = 0 )



Hình 160 : giản đồ thời gian bộ nhớ bên ngoài (SRWn1 = 0 , SRWn0 = 0 )



Note: 1. The ALE pulse in the last period (T4-T7) is only present if the next instruction accesses the RAM (internal or external).

## Các chỉ số thông thường

Các đồ thị dưới đây chỉ ra các xử lý thông thường . Các hình này không được kiểm tra trong suốt quá trình sản xuất . Tất cả các phép đo dòng điện tốn hao được tiến hành với tất cả các chân I/O được cấu hình như là các đầu ra và với các xung lên bên trong (Pull – ups) đã kích hoạt . Một máy phát sóng dạng sin với đầu ra rail – to – rail được sử dụng như là một nguồn phát xung nhịp .

Công suất tốn hao trong chế độ Power – down thì phụ thuộc vào sự lựa chọn xung nhịp

Các dòng điện tốn hao thì là 1 hàm của nhiều biến tỉ lệ như là : điện áp hoạt động , tần số điều khiển , tải trên các chân I/O , tốc độ chuyển mạch của các chân I/O , đoạn mã được thực thi và nhiệt độ làm việc , các hệ số ưu thế đang điều khiển điện áp và tần số

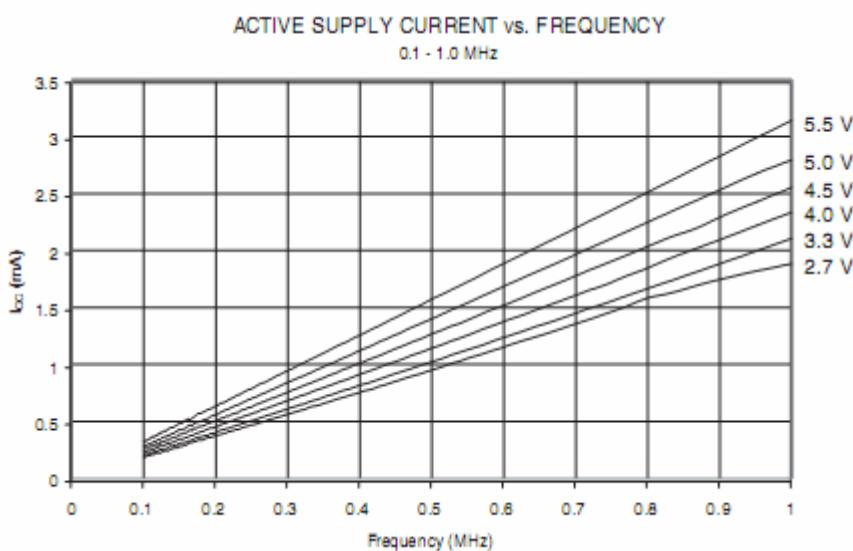
Dòng điện đã kéo từ các chân tải dung kháng có thể được ước lượng (cho một chân) như là  $C_L \cdot V_{CC} \cdot f$  ở đó  $C_L$  = điện dung tải ,  $V_{CC}$  = điện áp hoạt động và  $f$  = tần số chuyển mạch trung bình của chân I/O

Các phần được kí hiệu quy ước tại những tần số cao hơn những giới hạn kiểm tra . Các phần thì không được đảm bảo cho các chức năng thông thường tại các tần số cao hơn thứ tự các mã hiển thị

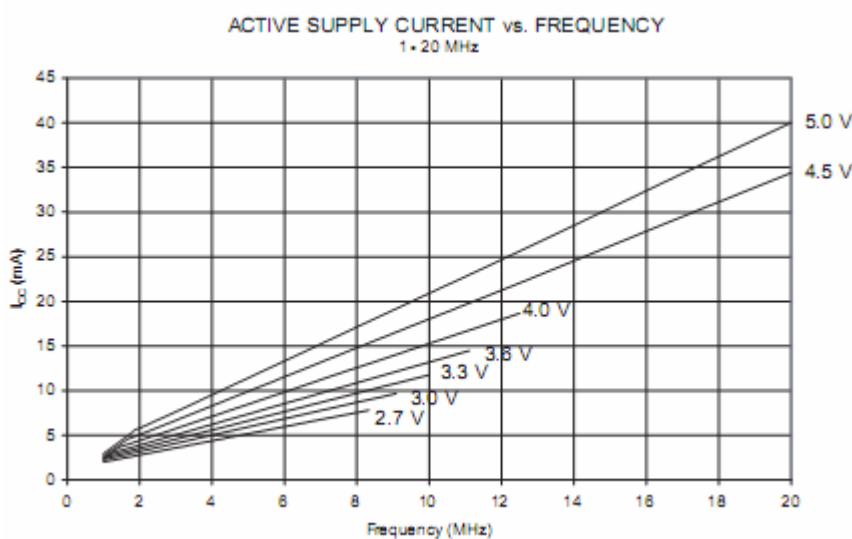
Sự khác nhau giữa dòng điện tổn hao trong chế độ Power – down với các timer watchdog đã kích hoạt và chế độ Power-down với các Timer Watchdog đã vô hiệu hóa đưa ra các dòng điện khác nhau được kéo bởi Timer Watchdog

### **Dòng điện nguồn cấp hoạt động (Active Supply Current )**

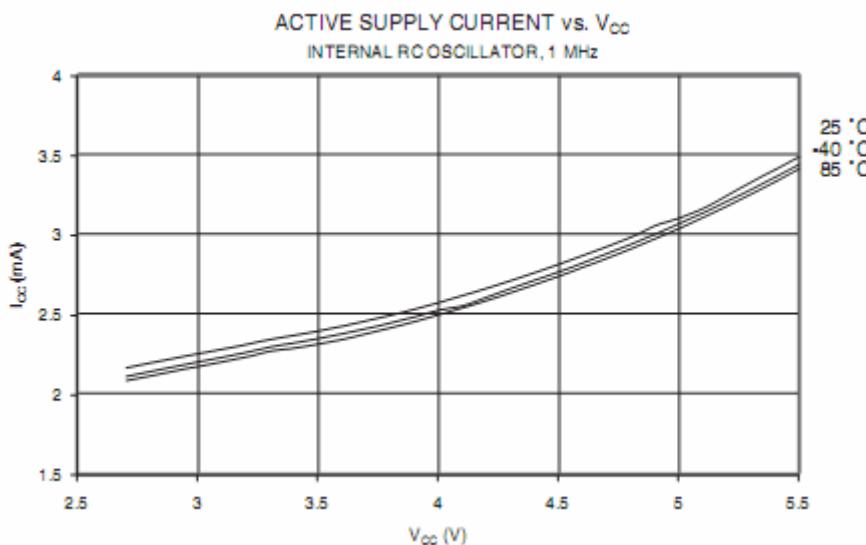
Hình 161 : dòng điện cung cấp hoạt động và tần số (0.1 – 1.0 MHz)



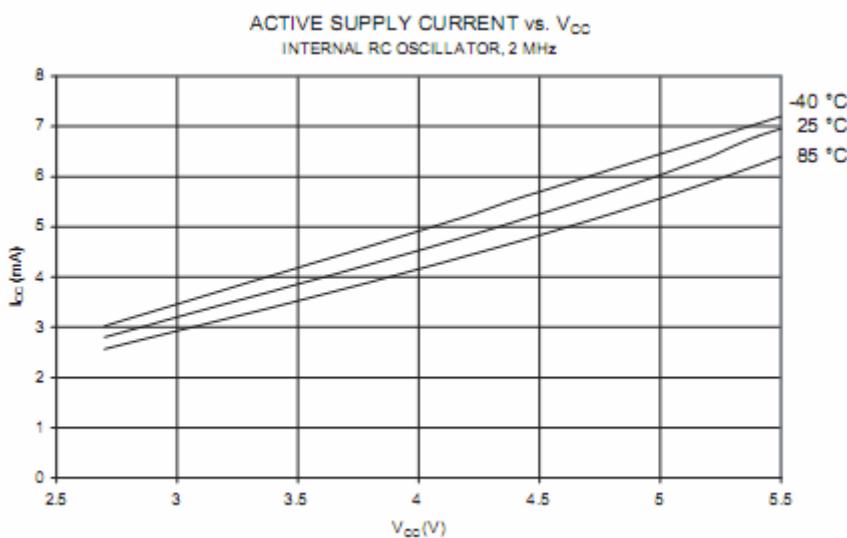
Hình 162 : dòng điện cung cấp hoạt động và tần số (1 – 20 MHz)



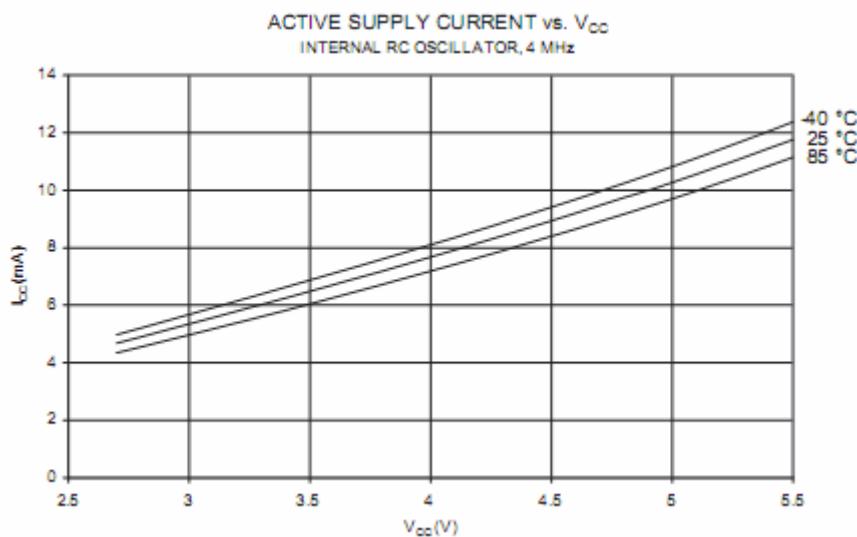
Hình 163 : dòng điện cung cấp hoạt động và  $V_{CC}$  (bộ tạo dao động RC bên trong , 1MHz)



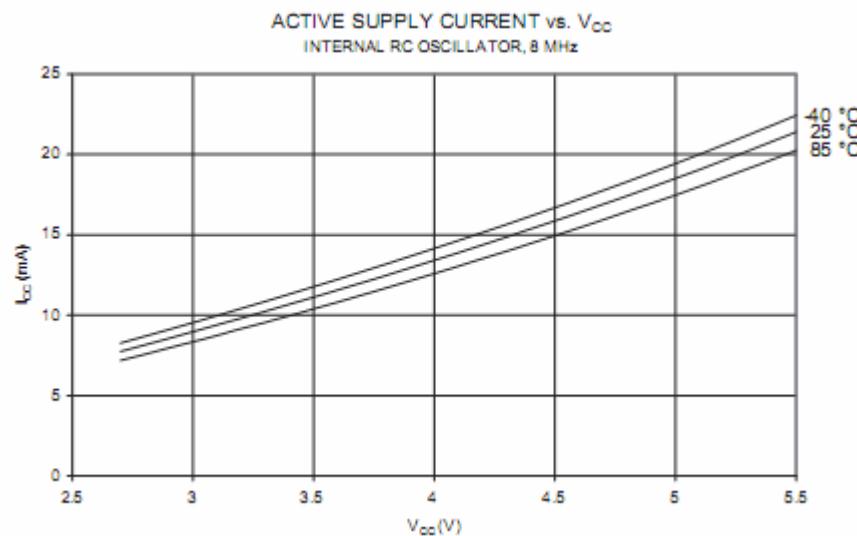
Hình 164 : dòng điện cung cấp hoạt động và V<sub>CC</sub> (bộ tạo dao động RC bên trong, 2MHz )



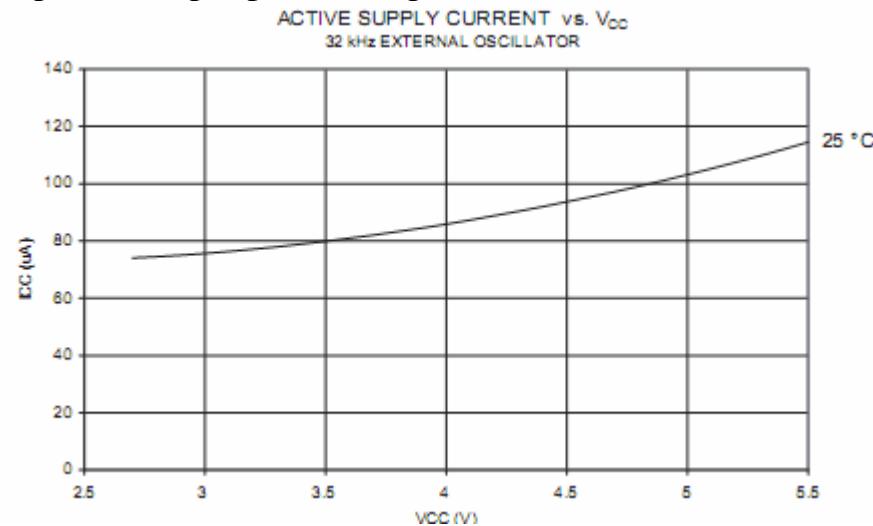
Hình 165 : dòng điện cung cấp tải bên trong và V<sub>CC</sub> (bộ tạo dao động bên trong 4MHz)



Hình 166 : dòng điện cung cấp tải hoạt động và V<sub>CC</sub> (bộ tạo dao động bên trong 8MHz)

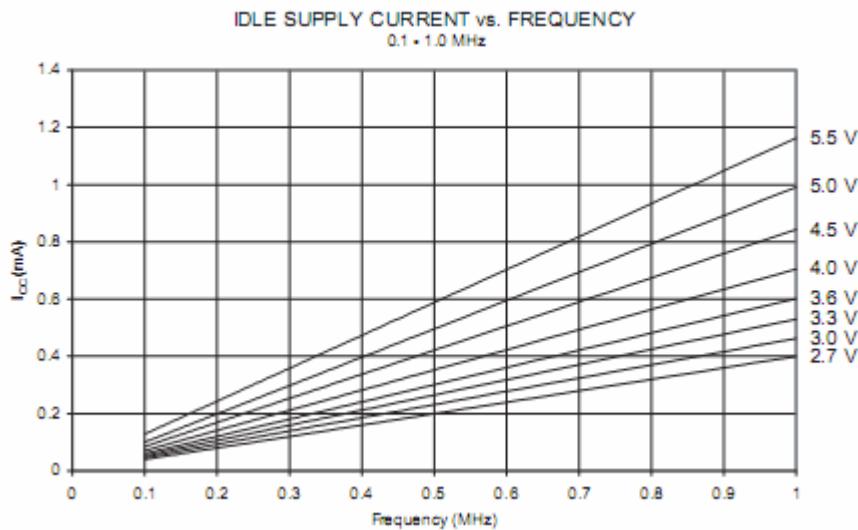


Hình 167 : dòng điện cung cấp hoạt động và V<sub>CC</sub> (32kHz bộ tạo dao động bên ngoài )

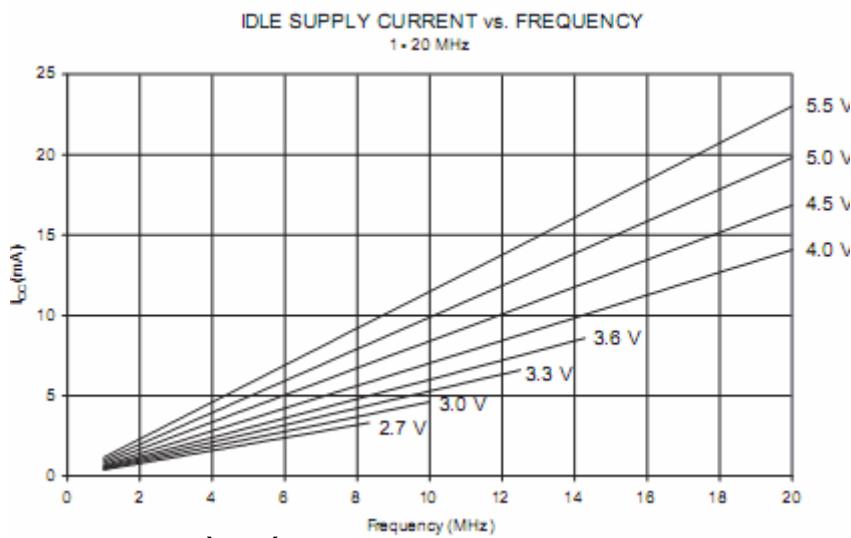


## Dòng điện cung cấp Idle (Idle supply current )

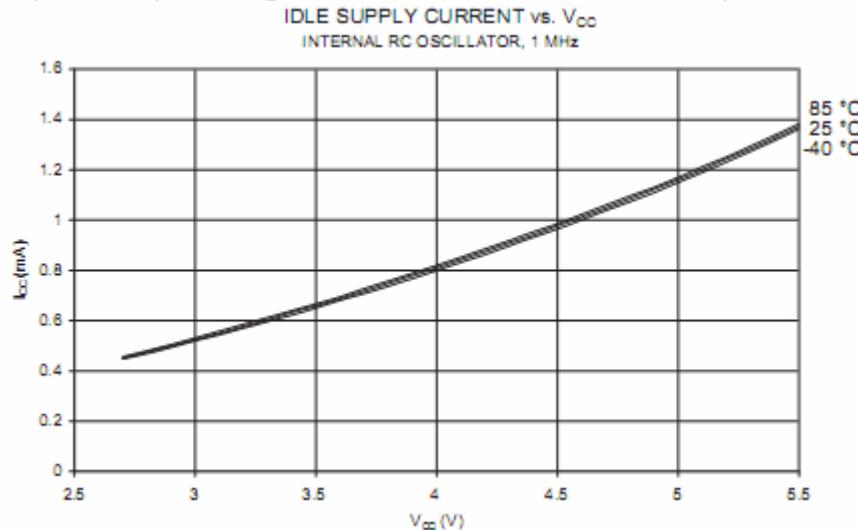
Hình 168 : dòng điện nguồn cấp Idle và tần số (0,1 – 1,0MHz)



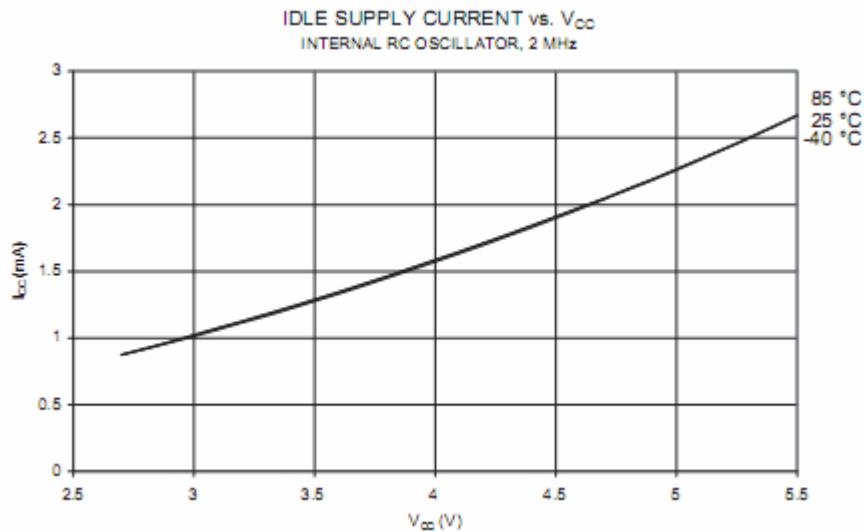
Hình 169 : dòng điện nguồn cấp Idle và tần số (1-20MHz)



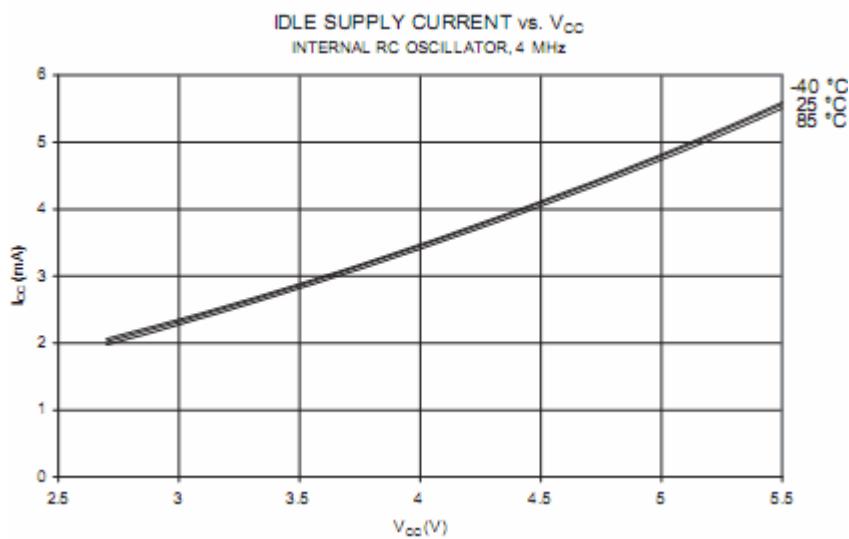
Hình 170 : dòng điện nguồn cấp Idle và V<sub>CC</sub> ( bộ tạo dao động RC bên trong , 1MHz)



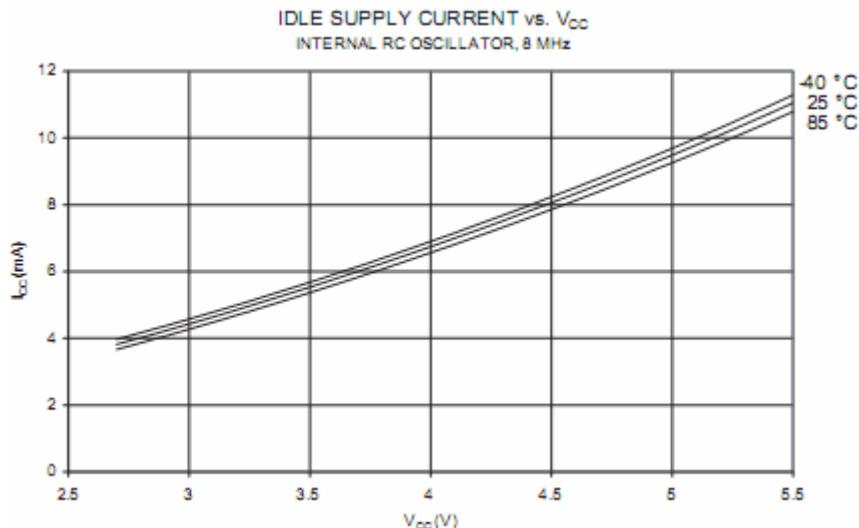
Hình 171 : dòng điện nguồn cấp Idle và V<sub>CC</sub>(bộ tạo dao động RC bên trong 2MHz)



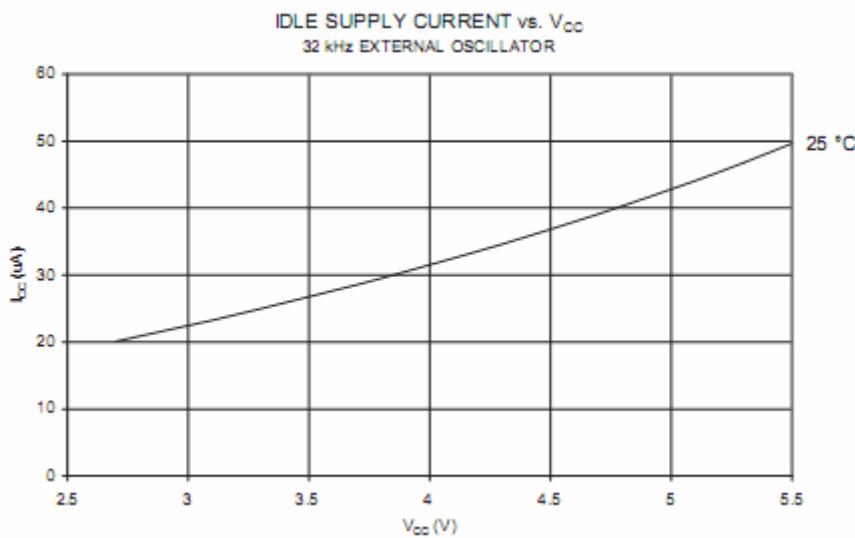
Hình 172 : dòng điện nguồn cấp Idle và V<sub>CC</sub>(bộ tạo dao động RC bên trong 4MHz)



Hình 173 : dòng điện nguồn cấp Idle và V<sub>CC</sub>(bộ tạo dao động RC bên trong 8MHz)

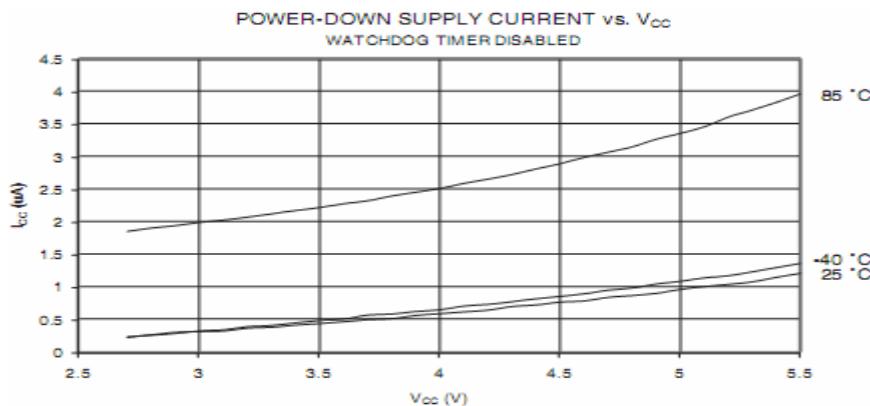


Hình 174 : dòng điện nguồn cấp Idle và V<sub>CC</sub> (bộ tạo dao động ngoài 32kHz )

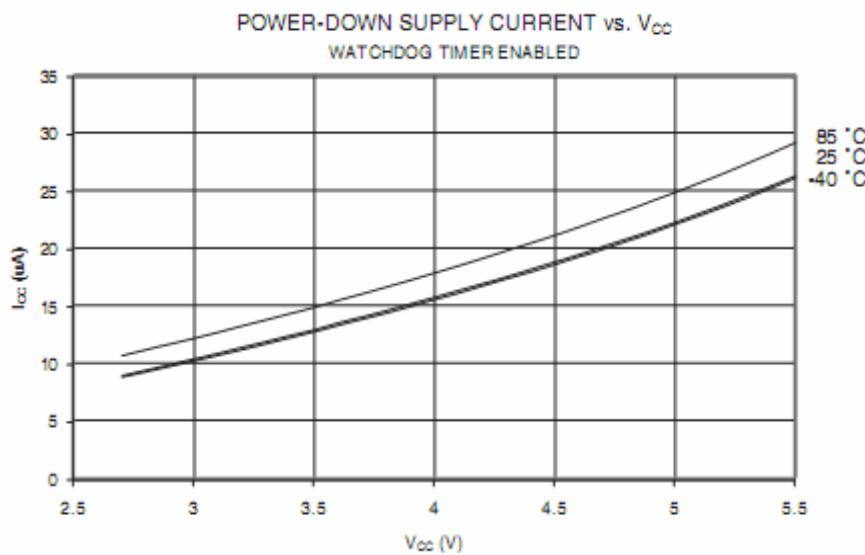


## Dòng điện nguồn cấp Power-down (Power –down Supply Current )

Hình 175 : dòng điện nguồn cấp Power – down và V<sub>CC</sub> (Timer watchdog bị vô hiệu hóa )

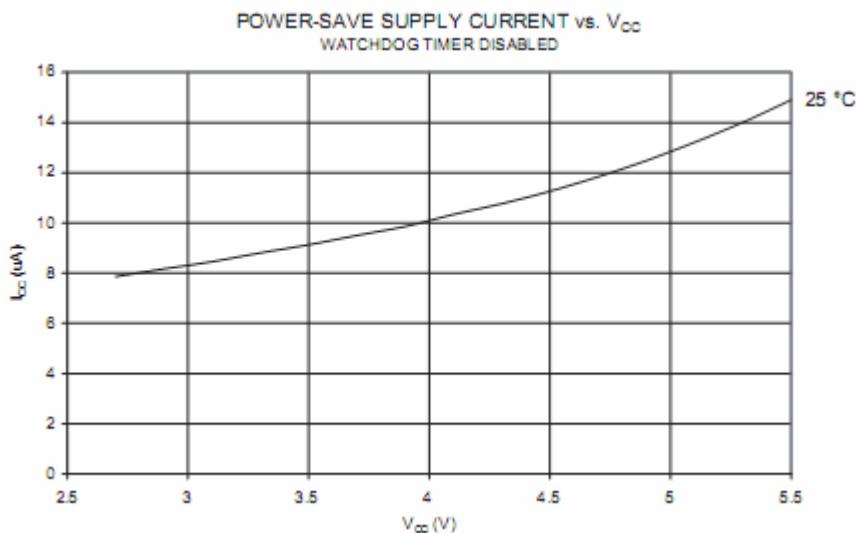


Hình 176 : dòng điện nguồn cấp Power – down và V<sub>CC</sub> (Timer watchdog đã kích hoạt )



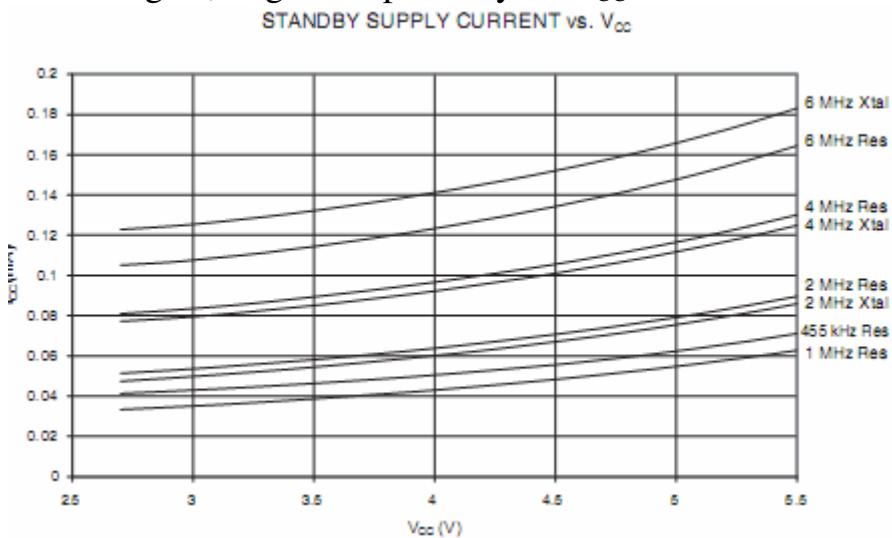
## Dòng điện nguồn cấp Power – save (Power – save supply current )

Hình 177 : dòng điện nguồn cấp Power – save và V<sub>CC</sub> (Timer watchdog bị vô hiệu hóa )

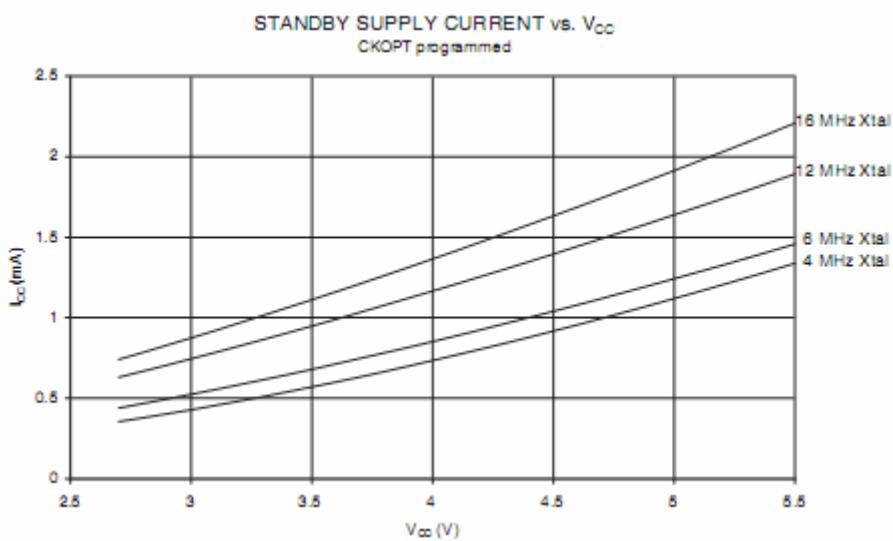


## Dòng điện nguồn cấp Standby (Standby supply current )

Hình 178 : Dòng điện nguồn cấp Standby và V<sub>CC</sub>

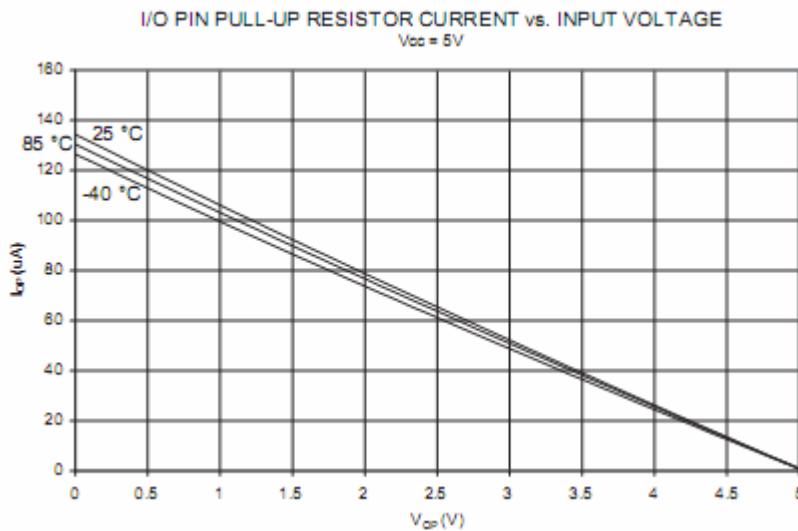


Hình 179 : Dòng điện nguồn cấp Standby và V<sub>CC</sub> (CKOPT đã lập trình )



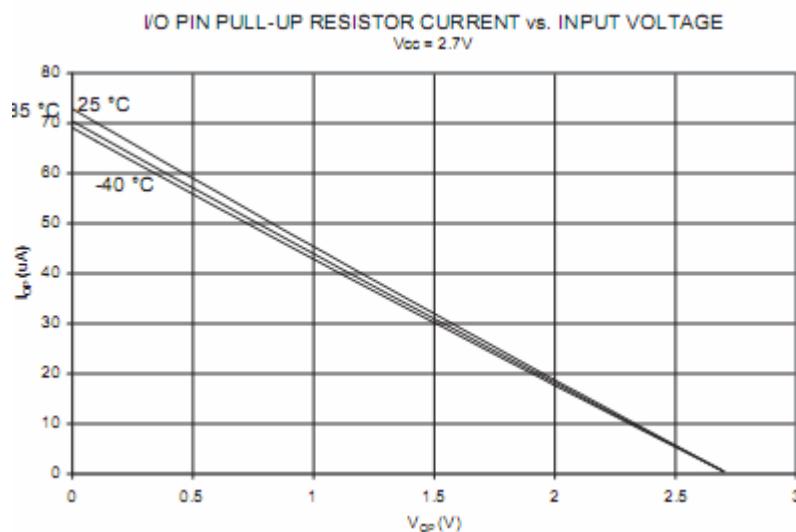
### Chân pull – up

Hình 180 dòng điện điện trở Pull – up chân I/O và điện áp đầu vào (V<sub>CC</sub> = 5V )



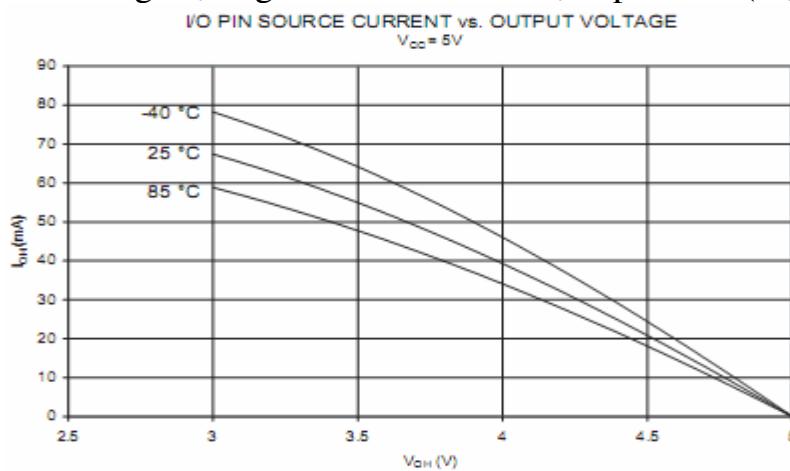
Hình 181 : dòng điện điện trở Pull – up chân I/O và điện áp đầu vào (V<sub>CC</sub> = 2,7V )

)

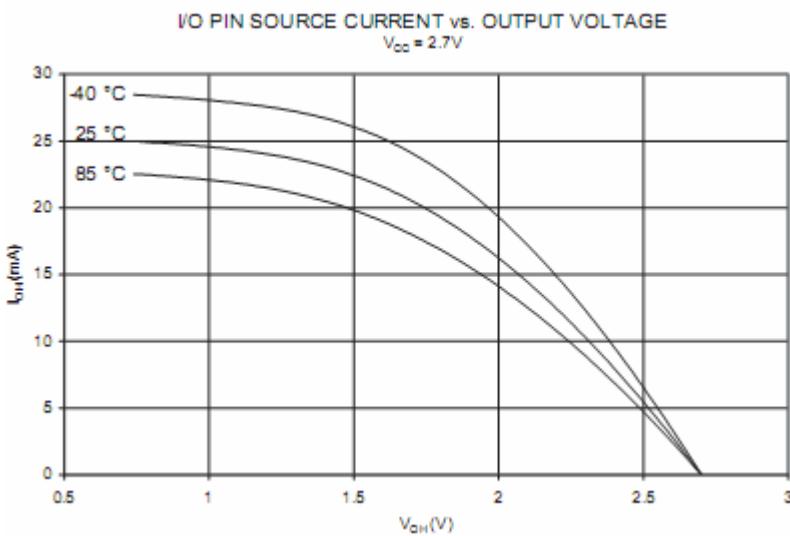


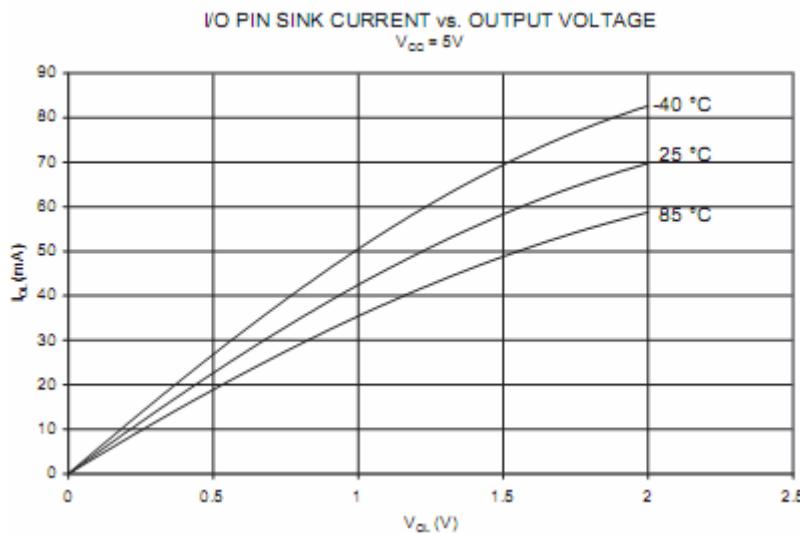
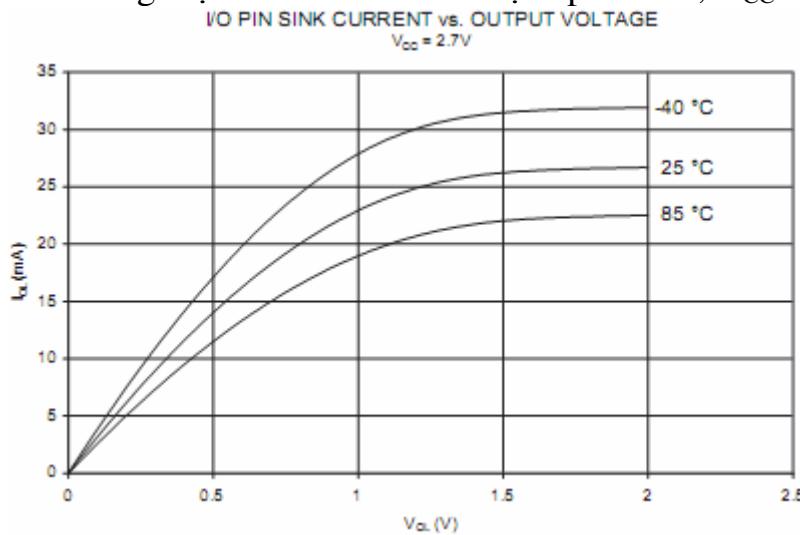
### Độ bền bộ điều khiển chân

Hình 182 : dòng điện nguồn chân I/O và điện áp đầu ra ( $V_{CC} = 5V$ )



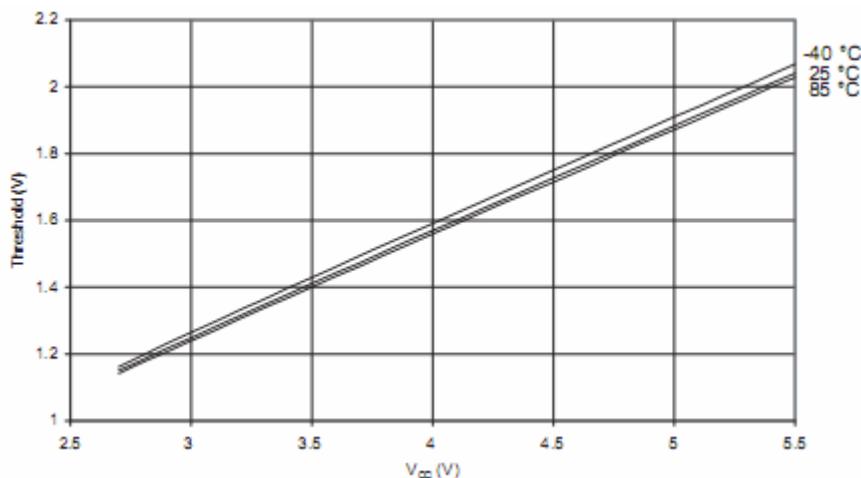
Hình 183 . dòng điện nguồn chân I/O và điện áp đầu ra ( $V_{CC} = 2.7V$ )



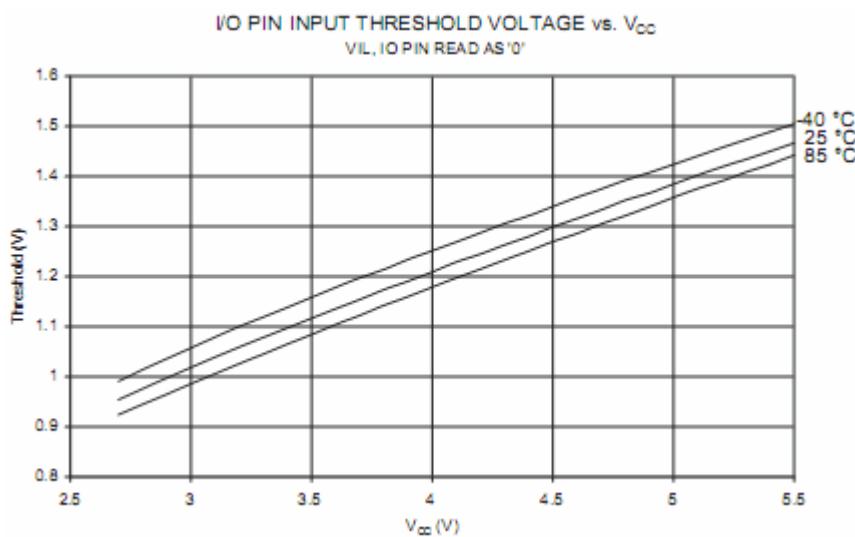
Hình 184 : dòng điện tản chân I/O và điện áp đầu ra ( $V_{CC} = 5V$ )Hình 185 : dòng điện tản chân I/O và điện áp đầu ra ,  $V_{CC} = 2.7V$ 

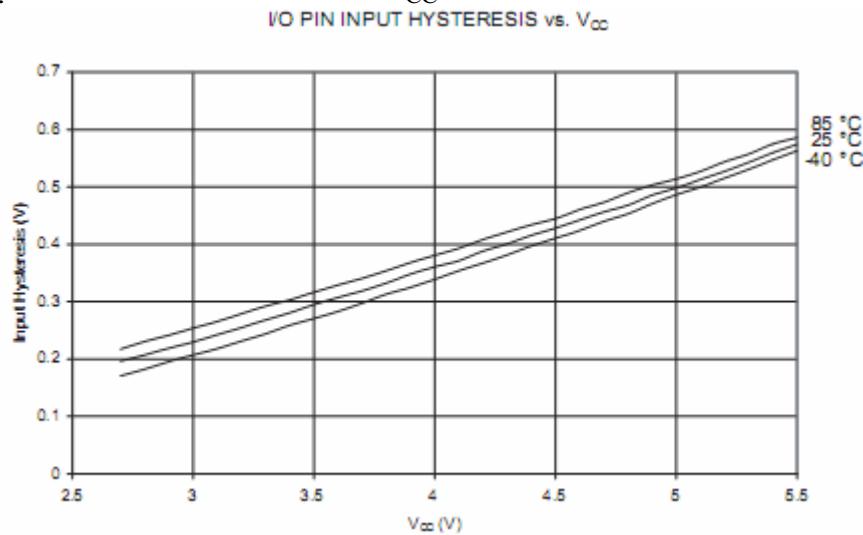
## Các ngưỡng giới hạn chân và sự trễ từ

Hình 186 : điện áp ngưỡng giới hạn đầu vào chân I/O và V<sub>cc</sub> (VIH , đọc chân I/O như là 1 )



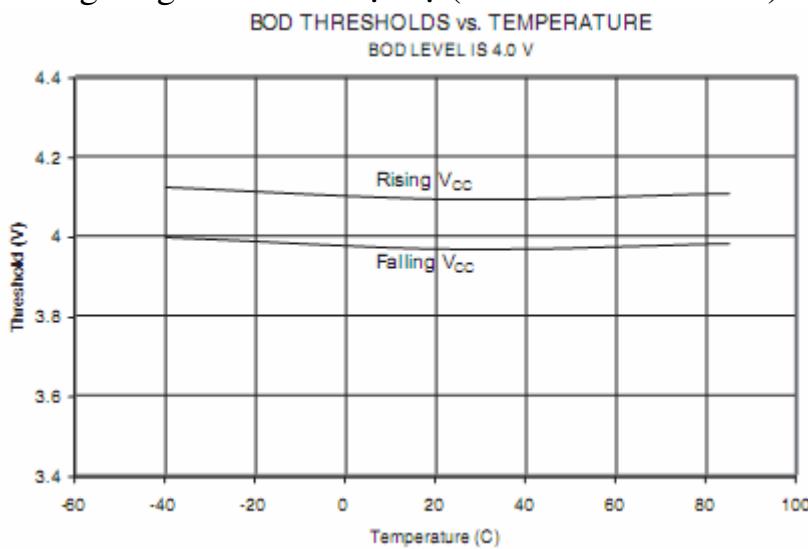
Hình 187 : điện áp ngưỡng đầu vào chân I/O và V<sub>CC</sub> (VIH , đọc chân I/O như là 0)



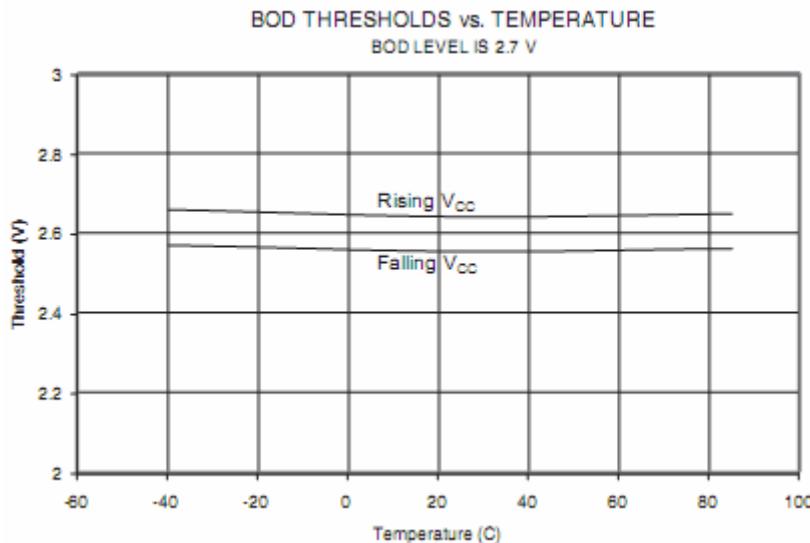
Hình 188 : độ trễ đầu vào chân I/O và V<sub>CC</sub>

### Nguõng BOD và bô so sánh tương tự Offset

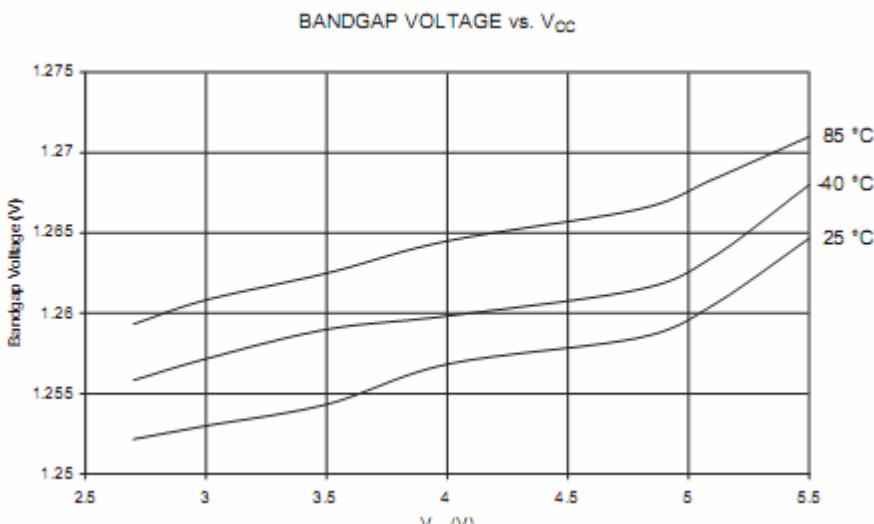
Hình 189 : nguõng BOD và nhiệt độ (BODLEVEL là 4 V )



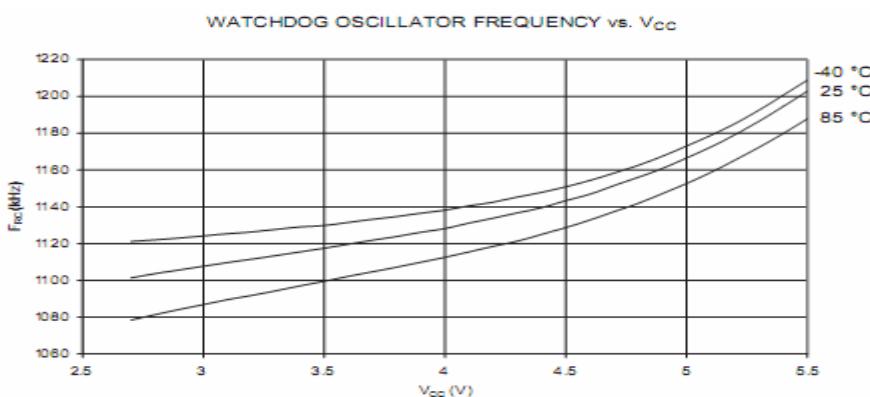
Hình 190 : ngưỡng BOD và nhiệt độ (BODLEVEL là 2.7V )



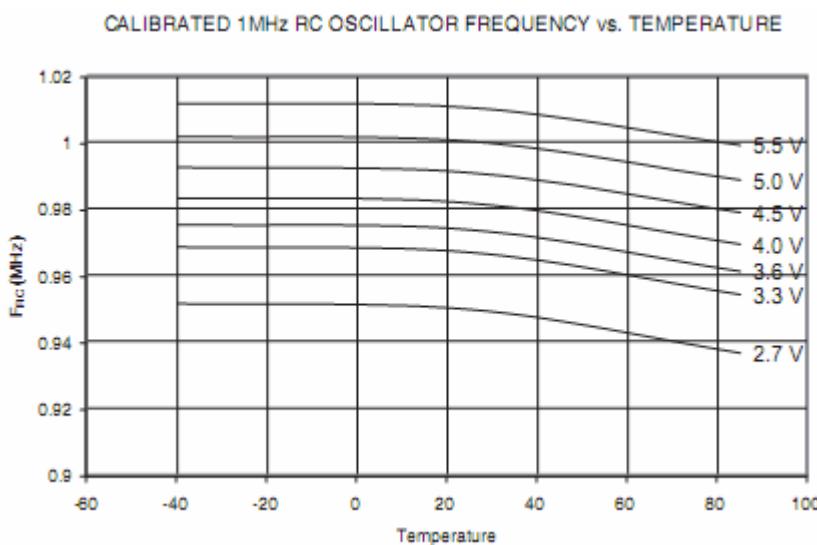
Hình 191 : điện áp bandgap và điện áp điều khiển



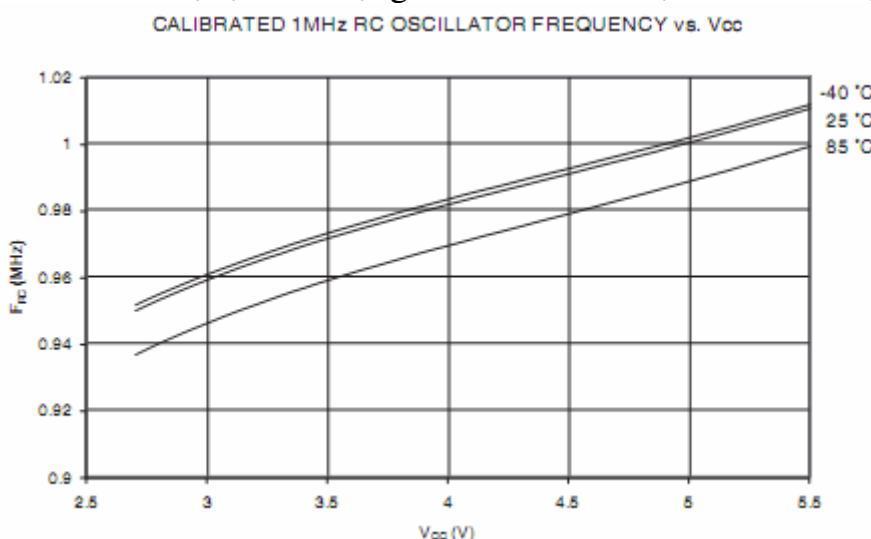
### Tốc độ của bộ tạo dao động bên trong

Hình 192 : tần số bộ tạo dao động watchdog và V<sub>CC</sub>

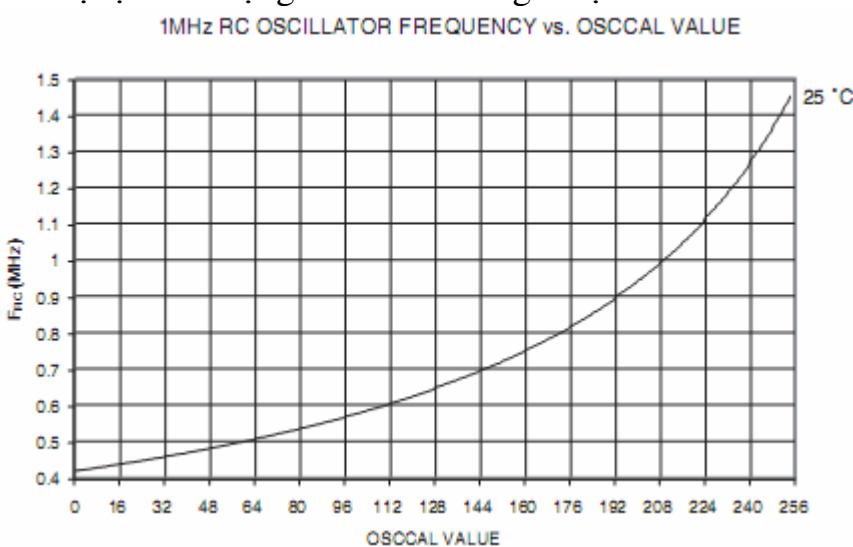
Hình 193 : tần số bộ tạo dao động RC 1MHz đã hiệu chỉnh và nhiệt độ



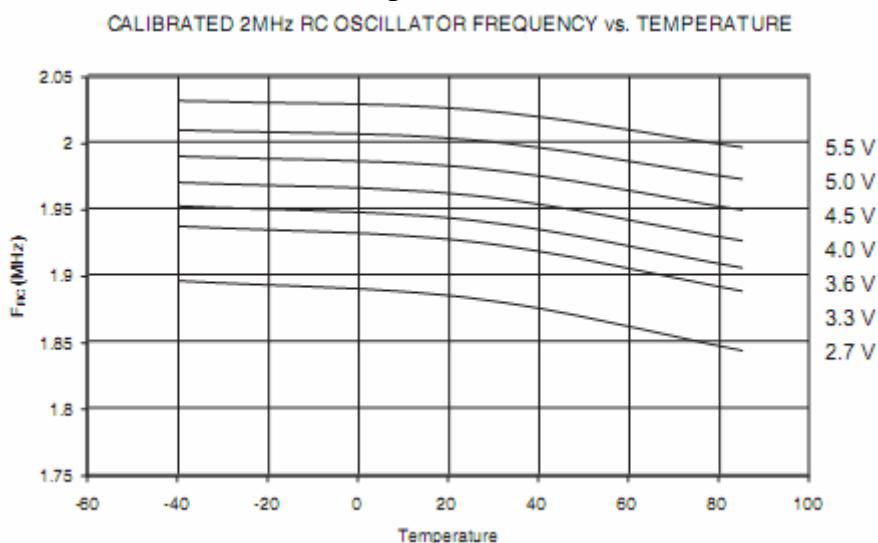
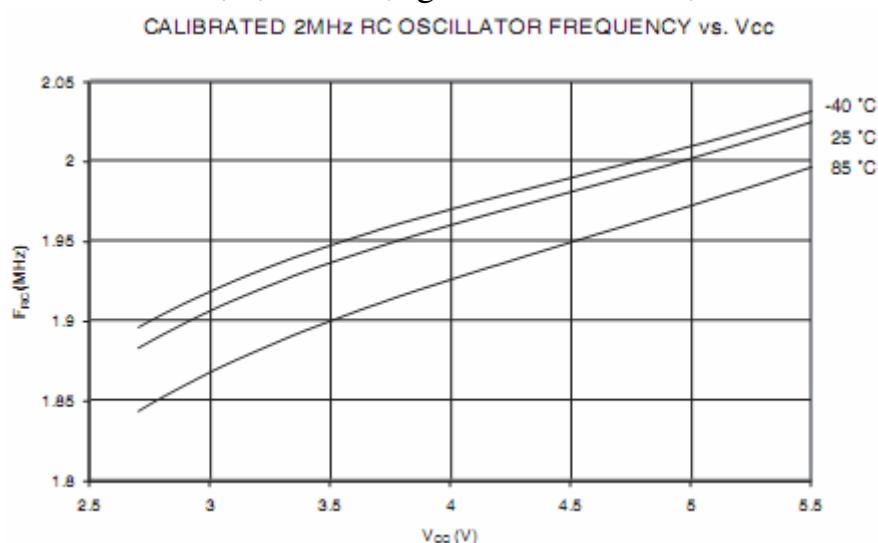
Hình 194 : tần số bộ tạo dao động RC 1MHz đã hiệu chỉnh và V<sub>CC</sub>



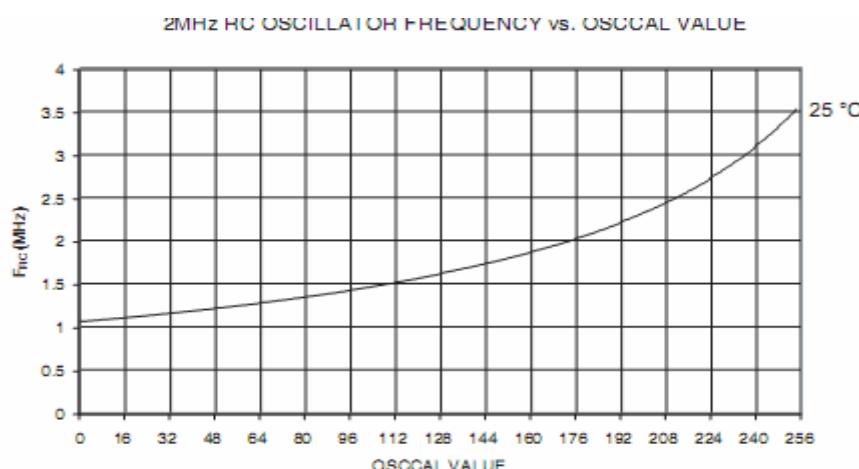
Hình 195 : tần số bộ tạo dao động RC 1MHz và giá trị OscCal



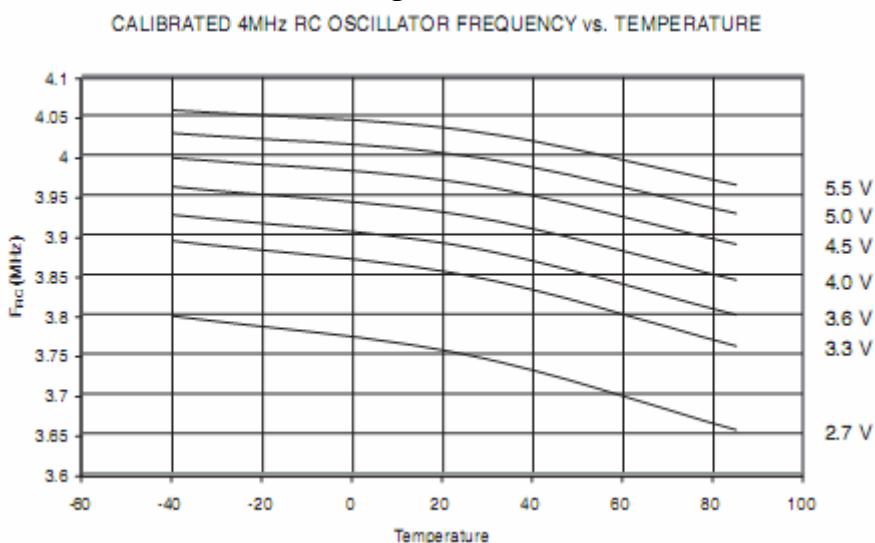
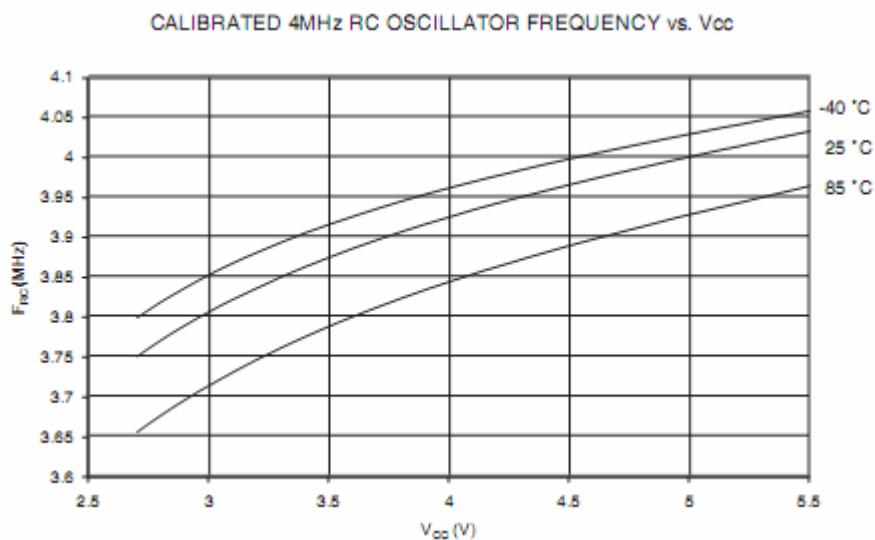
Hình 196 : tần số bộ tạo dao động RC 2MHz đã hiệu chỉnh và nhiệt độ

Hình 197 : tần số bộ tạo dao động RC 2MHz đã hiệu chỉnh và V<sub>CC</sub>

Hình 198 : tần số bộ tạo dao động RC 2MHz và giá trị OscCal

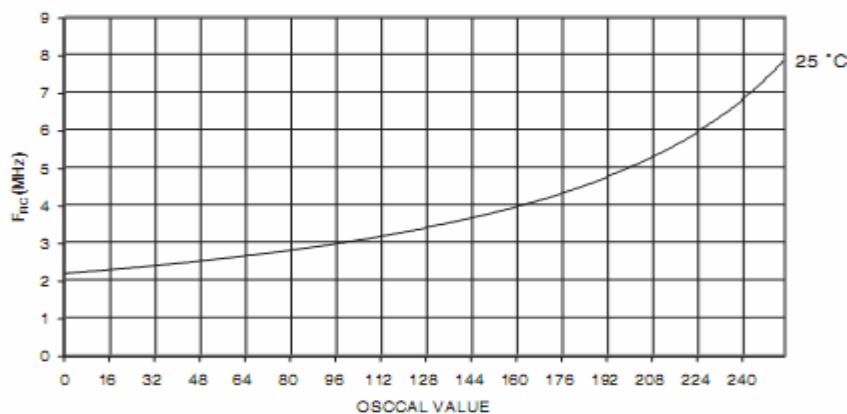


Hình 199 : tần số bộ tạo dao động RC 4MHz đã hiệu chỉnh và nhiệt độ

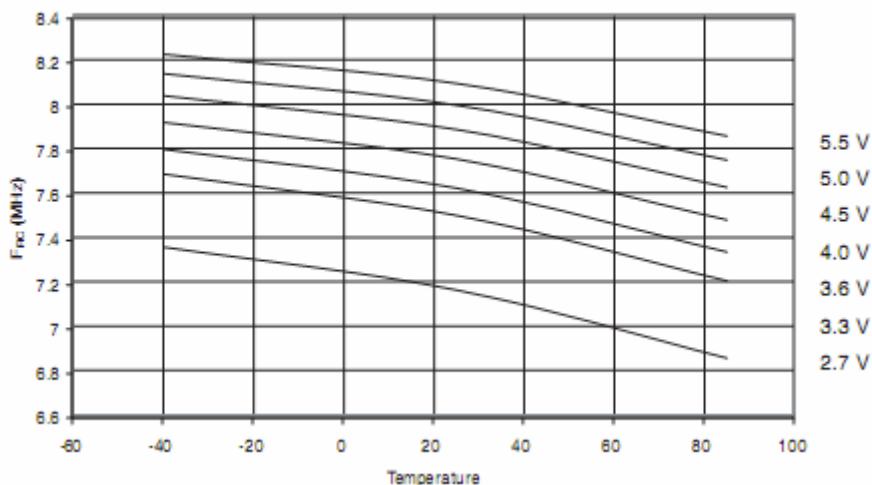
Hình 200 : tần số bộ tạo dao động RC 4MHz đã hiệu chỉnh và V<sub>CC</sub>

Hình 201 : tần số bộ tạo dao động RC 4MHz và giá trị OscCal

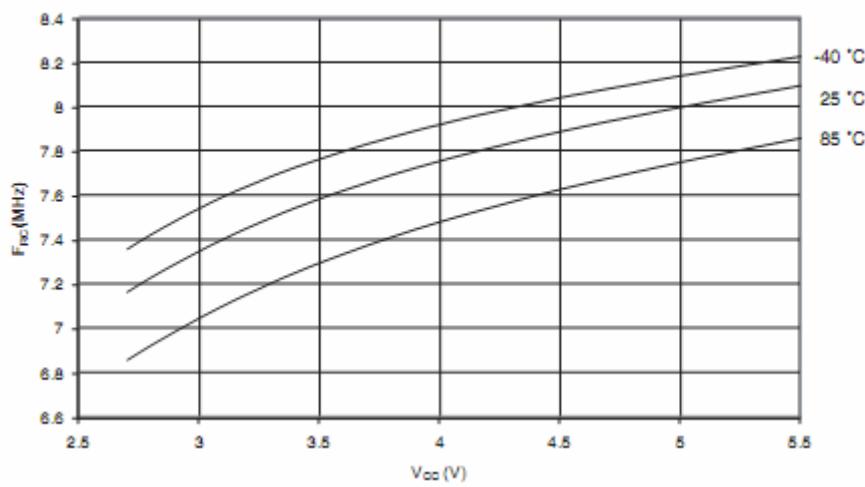
4MHz RC OSCILLATOR FREQUENCY vs. OSCCAL VALUE



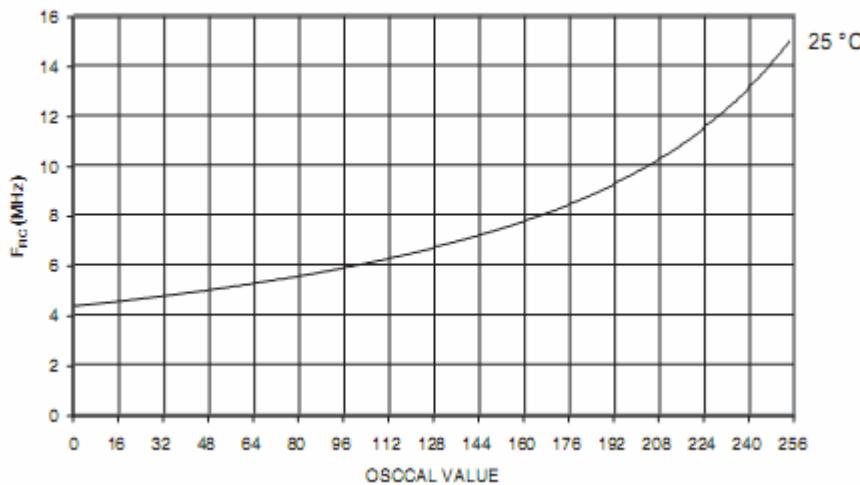
Hình 202 : tần số bộ tạo dao động RC 8MHz đã hiệu chỉnh và nhiệt độ  
**CALIBRATED 8MHz RC OSCILLATOR FREQUENCY vs. TEMPERATURE**



Hình 203 : tần số bộ tạo dao động RC 8MHz đã hiệu chỉnh và V<sub>CC</sub>  
**CALIBRATED 8MHz RC OSCILLATOR FREQUENCY vs. V<sub>CC</sub>**

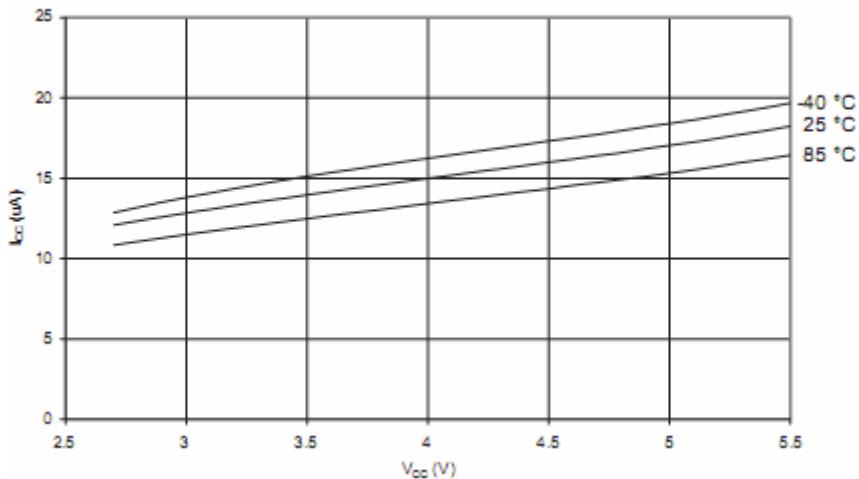


Hình 204 : tần số bộ tạo dao động RC 8MHz và giá trị OscCal  
**8MHz RC OSCILLATOR FREQUENCY vs. OSCCAL VALUE**

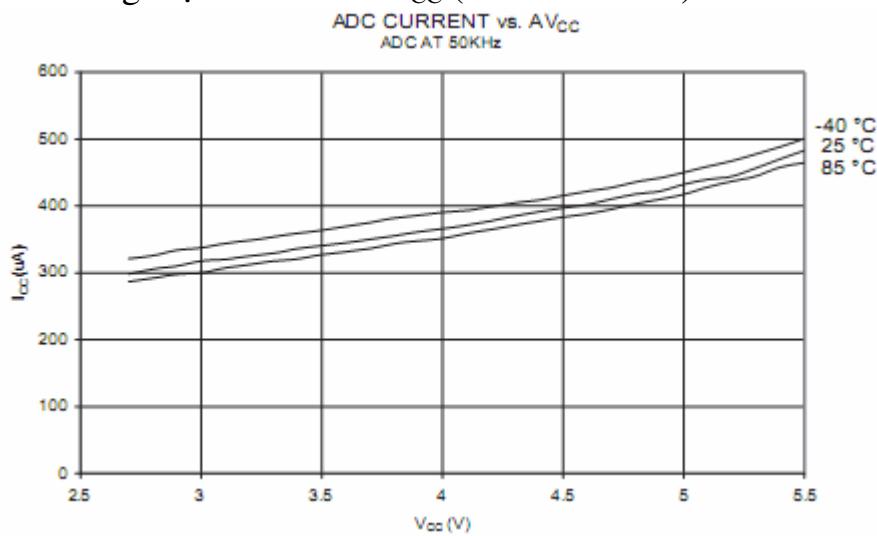


## Dòng điện tổn hao của các thành phần ngoại vi

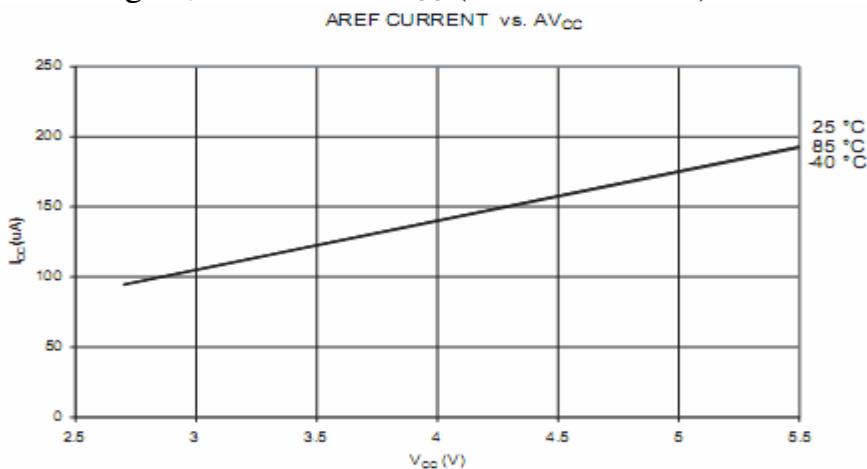
Hình 205 : dòng điện máy dò yếu nguồn điện và V<sub>CC</sub>  
 BROWNOUT DETECTOR CURRENT vs. V<sub>CC</sub>

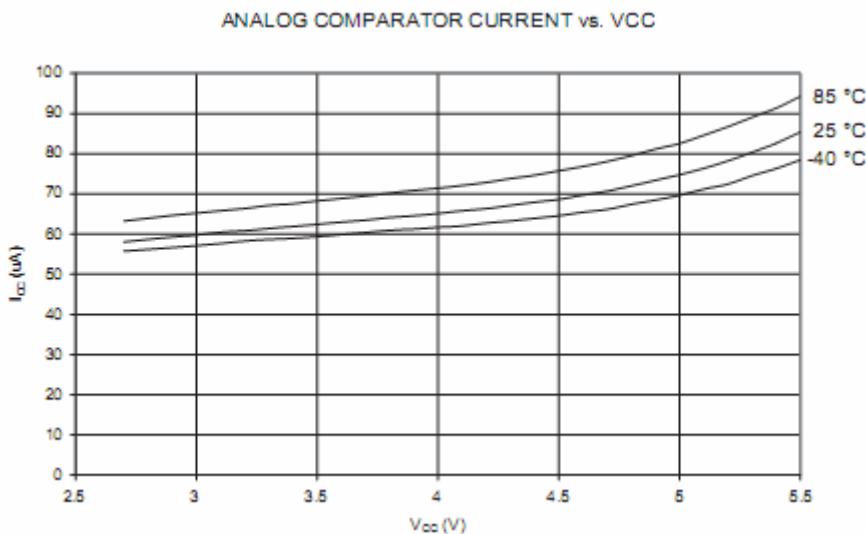
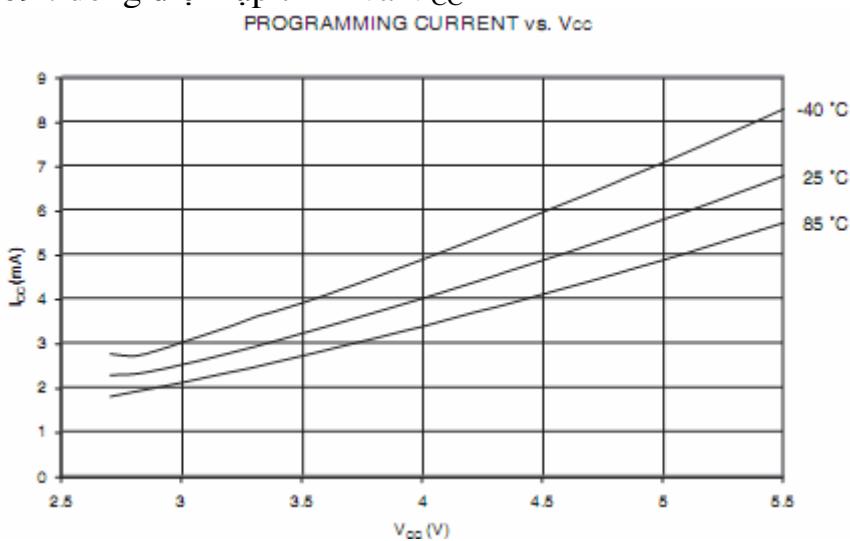


Hình 206 : dòng điện ADC và AV<sub>CC</sub> (ADC ở 50kHz)

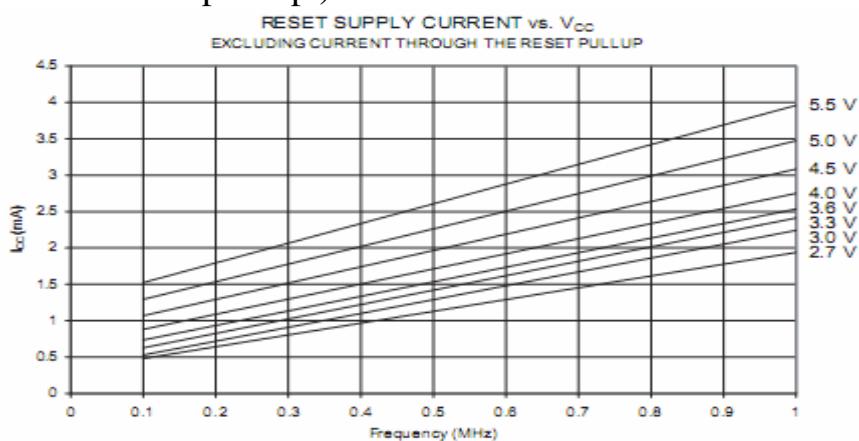


Hình 207 : dòng điện ADC và AV<sub>CC</sub> (ADC ở 1 MHz )

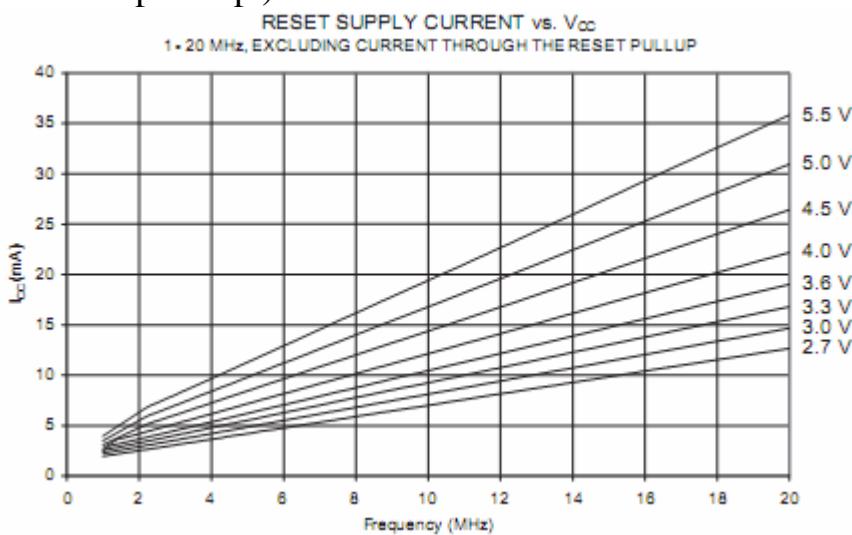


Hình 208 : dòng điện bộ so sánh tương tự và V<sub>CC</sub>Hình 209 : dòng điện lập trình và V<sub>CC</sub>

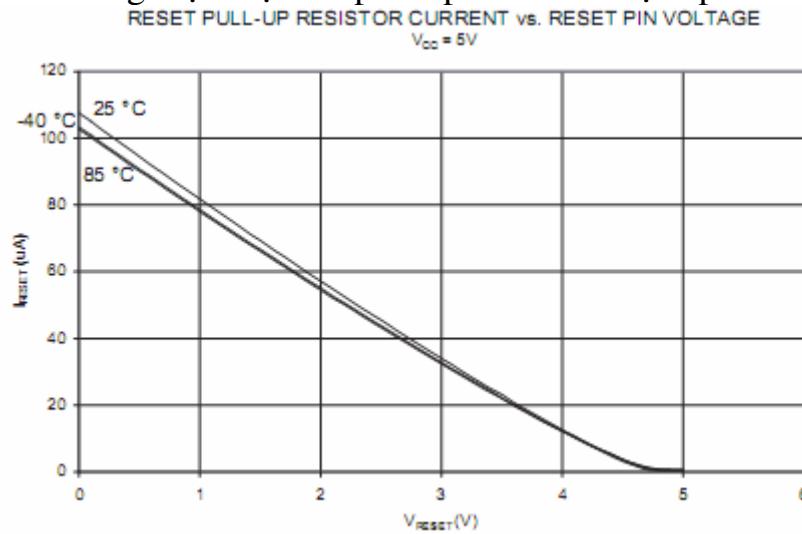
### Dòng điện tốn hao trong khi reset và độ rộng xung reset

Hình 210 : dòng điện nguồn cấp Reset và V<sub>CC</sub> (0,1 – 1,0 MHz k, ;bao gồm dòng điện thông qua chân reset pull-up )

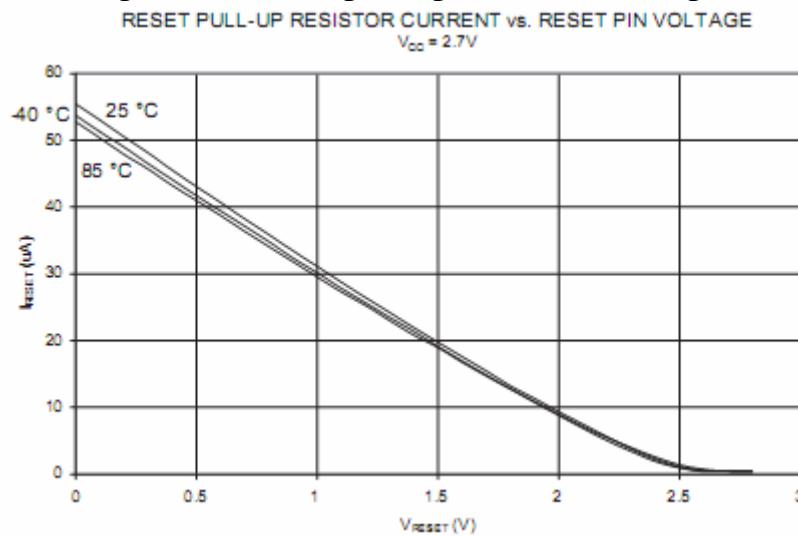
Hình 211 : dòng điện nguồn cấp Reset và V<sub>CC</sub> (1 – 20 MHz , bao gồm dòng điện thông qua chân Reset pull- up )

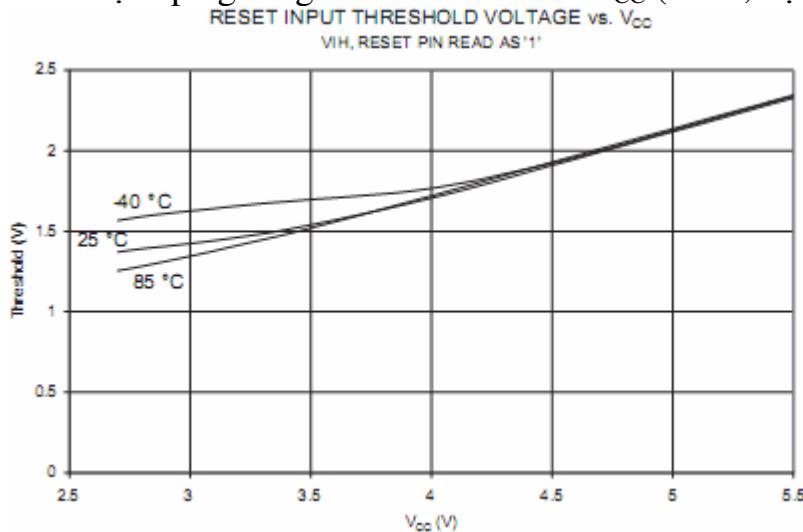
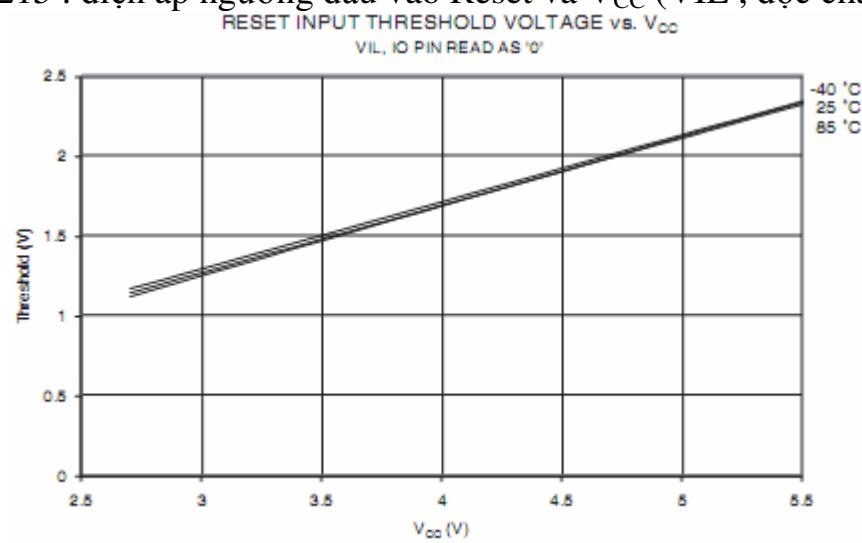
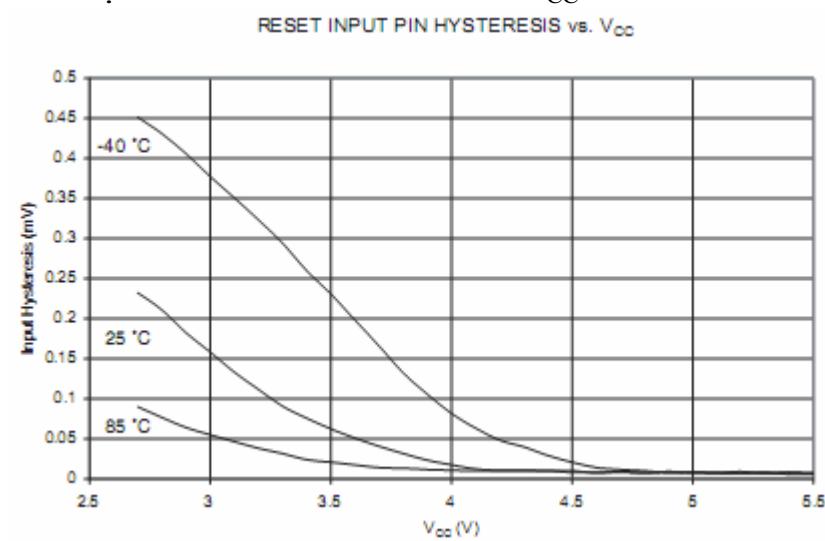


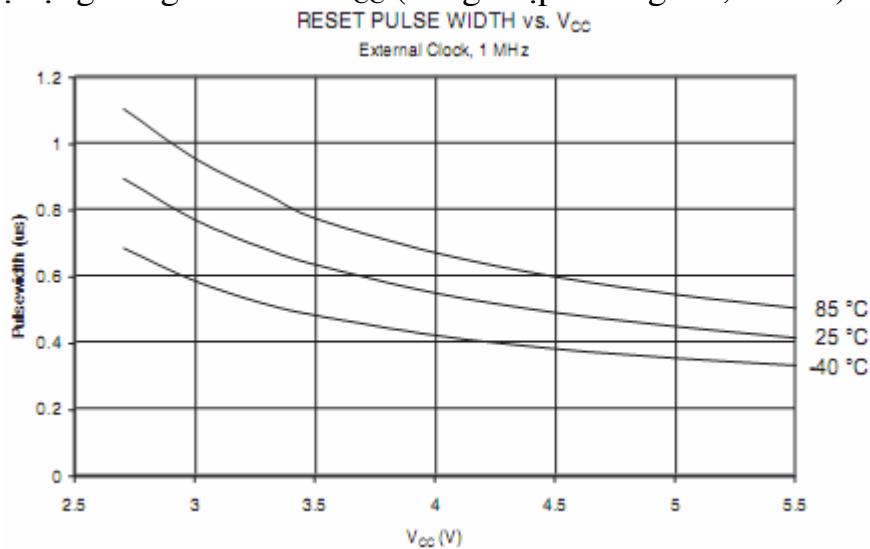
Hình 212 : dòng điện điện trở pull-up Reset và điện áp chân reset (V<sub>CC</sub> = 5.0 V )



Hình 213 : dòng điện điện trở pull-up Reset và điện áp chân reset (V<sub>CC</sub> = 2.7 V )



Hình 214 : điện áp ngưỡng đầu vào Reset và V<sub>CC</sub> (VIH , đọc chân Reset là 1)Hình 215 : điện áp ngưỡng đầu vào Reset và V<sub>CC</sub> (VIL , đọc chân Reset là 0)Hình 216 : độ trễ chân đầu vào Reset và V<sub>CC</sub>

Hình 217 : độ rộng xung Reset và V<sub>CC</sub> (xung nhịp bên ngoài , 1MHz)

Bảng kê chi tiết thanh ghi tham khảo tai lieu file PDF

## Mục lục

I . Đặc điểm tính năng của AVR 128 .....	1
II . Cấu hình chân .....	2
III. Lõi AVR.....	8
IV. Các bộ nhớ của AVR .....	18
V. Xung nhịp hệ thống và lựa chọn xung nhịp .....	40
VI .Quản lí nguồn điện và các chế độ SLEEP .....	48
VII . Điều khiển hệ thống và RESET .....	53
VIII . Các ngắt .....	65
IX. Các cổng vào ra .....	80
X. Các ngắt ngoài .....	96
XI. Timer/Counter8 bit với PWM .....	99
XII.Timer/Counter 16bit(1 , 3) .....	119
XIII. Các bộ đếm gộp trước của TC1, TC2 , TC3 .....	154
XIV. Timer/Counter 8 bit với PWM.....	156
XV. Khởi module so sánh đầu ra (OCM1C2) .....	173
XVI . Giao diện ngoại vi nối tiếp SPI.....	177
XVII . USART .....	185
XVIII. Giao diện 2 dây tuần tự TWI.....	215
XX. Bộ so sánh tương tự .....	246
XXI . Bộ chuyển đổi tương tự sang số .....	250
XXII . Giao diện JTAG và hệ thống hiệu chỉnh lỗi trên chip.....	268
XXIII . Hỗ trợ bộ tải chương trình lập trình READ WHILE WRITE.....	293
XXIV. Các đặc tính điện .....	340

14/3/2010

