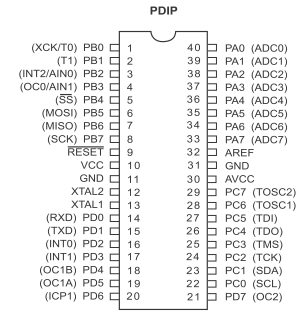
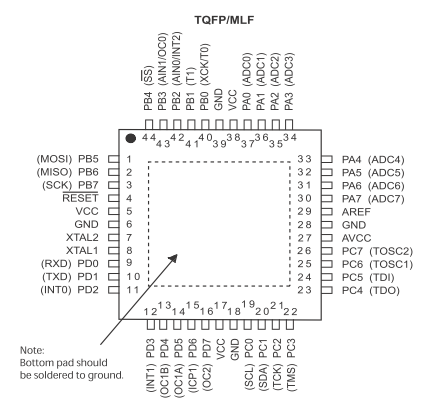
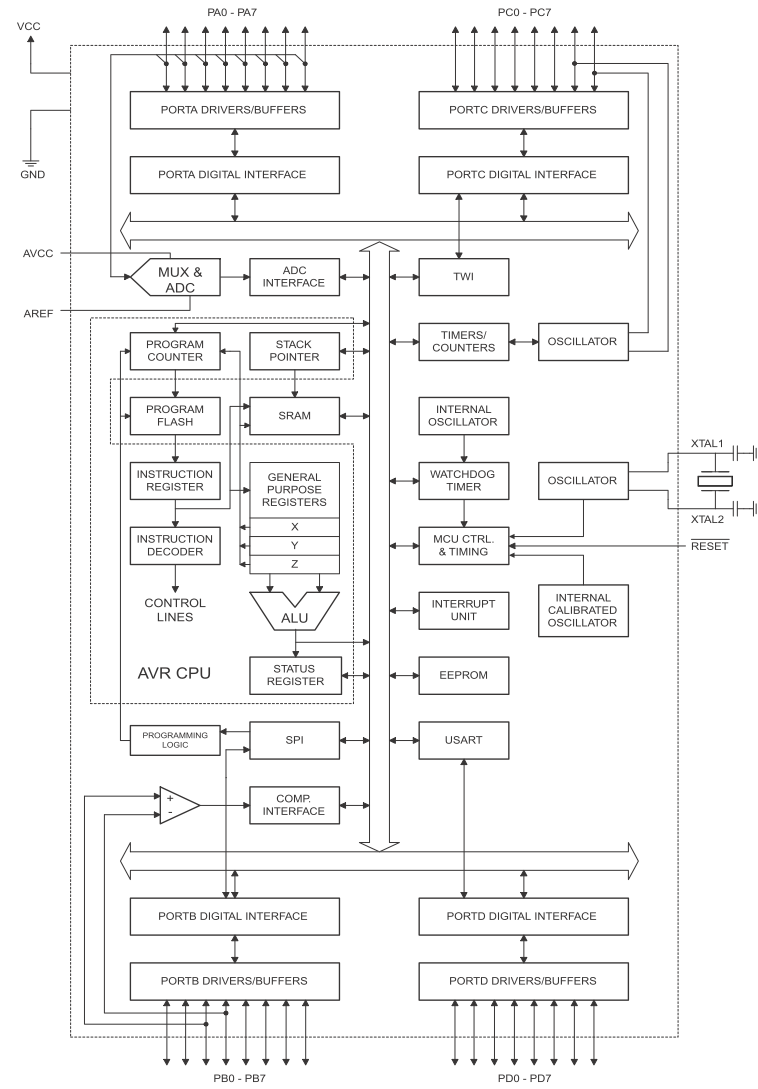
* Vi điều khiển Atmega32:
  + Là vi điều khiển 8-bit CMOS công suất tiêu thụ thấp dựa trên cấu trúc RISC AVR. Bằng cách thực hiện các lệnh mạnh trong một chu kỳ đồng hồ. Atmega32 đạt được tốc độ xấp xỉ 1MIPS trên 1MHz cho phép người thiết kế tối ưu công suất tiêu thụ với tốc độ sử lý.
* Các tính năng:
  + Cấu trúc RISC
    - Hỗ trợ 131 lệnh
    - 32x8 thanh ghi dùng chung
    - Tốc độ 16 MIPS với thạch anh 16 MHz
  + Bộ nhớ
    - 32K Byte bộ nhớ Flash
    - 1K Byte EEPROM
    - 2K Byte SRAM
    - Hỗ trợ lập trình ngay trên mạch với chương trình mồi
    - Có các bít khóa bảo mật
  + Giao diện JTAG
    - Khả năng quét biên theo chuẩn JTAG
    - Hỗ trợ gỡ lỗi trên chip
    - Lập trình bộ nhớ Flash, EEPROM, các bít khóa qua giao diện JTAG
  + Các tính năng ngoại vi
    - Hai bộ đếm/định thời 8 bit
    - Một bộ đếm / định thời 16 bit
    - Bộ đếm thời gian thực với bộ giao động riêng
    - Bốn kênh PWM
    - 8 kênh ADC 10 bit
    - Giao tiếp I2C, USART, SPI.
    - Bộ so sánh Analog trên chíp
  + Các tính năng khác
    - Có bộ giao động RC bên trong
    - Ngắt trong và ngắt ngoài
    - Sáu chế độ ngủ
  + Đóng gói I/O
    - 32 đường I/O lập trình được
    - 40 chân PDIP
  + Điện áp hoạt động
    - 2.7 – 5.5 V với Atmega32L
    - 4.5 – 5.5 V với Atmega32
  + Tốc độ
    - 0 - 8 MHz với Atmega32L
    - 0 - 16 MHz với Atmega32
* Đóng gói:





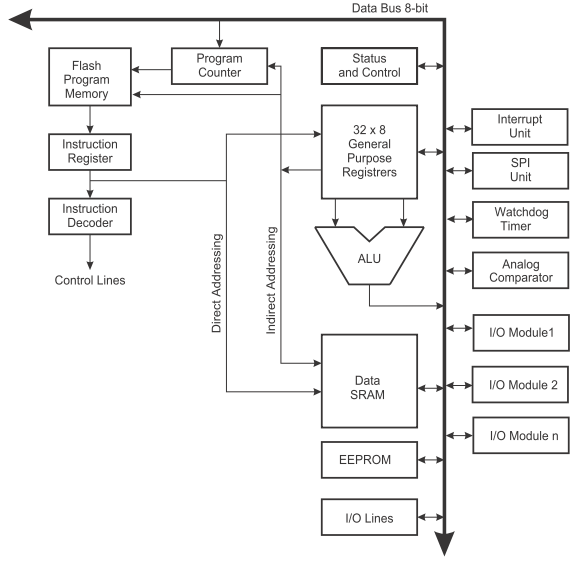
Hình : Sơ đồ đóng gói Atmega32

* Sơ đồ khối:



Hình : Sơ đồ khối Atmega32

* Mô tả chân:
  + VCC: Điện áp cung cấp.
  + GND: Nối đất.
  + PortA, PortB, PortC, PortD: Có thể được cấu hình như các cổng cổng I/O 8-bit hai hướng thông thường hoặc cấu hình để sử dụng các chức năng đặc biệt khác . Các chân của Port có thể được nối với các điện trở kéo lên bên trong (lựa chọn cho từng bit).
  + Reset: Là đầu vào. Khi đưa một mức điện áp thấp vào chân này có độ dài lớn hơn một xung sẽ reset hệ thống.
  + XTAL1, XTAL2: Các đầu vào dao động.
  + AVCC: Điện áp cấp cho các bộ ADC. Thường được nối với VCC qua một bộ lọc thông thấp.
  + AREF: Điện áp tham chiếu cho các bộ biến đổi ADC.
* Lõi CPU AVR

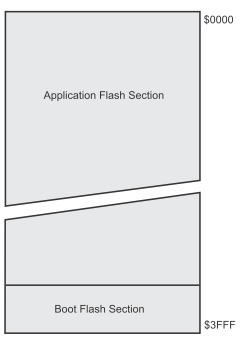


Hinh : Lõi CPU AVR

* + 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 được thực thi với 1 cấp sử lí liên lệnh đơn . Trong khi lệnh đang được xử lí thì lệnh tiếp theo được nạp tiếp từ bộ nhớ chương trình . Khái niệm này kích hoạt lệnh để thực thi trong mỗi chu kì xung nhịp đồng hồ . Bộ nhớ chương trình là bộ nhớ flash có thể lập trình lại được ở trong hệ thống .
  + 6 trong 32 thanh ghi chung 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 các thanh ghi 16 bit X , Y , Z.
  + 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 ngắt 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 gán trong vùng dữ liệu chung SRAM , và do đó kích thước ngăn xếp ( stack ) chỉ bị giới hạn bởi độ lớn của SRAM và độ sử dụng của SRAM. Tất cả các chương trình người dùng phải được khởi tạo SP ( stack poiter) trong thủ tục reset (trước khi chương trình con hoặc các ngắt được thực thi ) . Con trỏ ngăn xếp ( SP- stack pointer ) được truy cập đọc/ghi ở trong không gian địa chỉ I/O. Dữ liệu SRAM 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 .
  + Tất cả các ngắt đều có một véc tơ ngắt riêng trong bảng vecto ngắt.

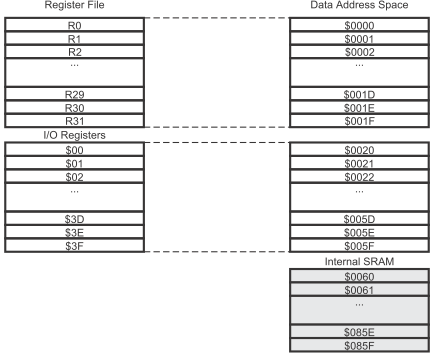
Các ngắt có quyền ưu tiên theo vị trí vector ngắt của nó . Vector ngắt có địa chỉ càng thấp thì ngắt càng được ưu tiên hơn.

* Các bộ nhớ:
  + Bộ nhớ Flash: 32K bytes được quản lý như 16K x 16 do hầu hết các lệnh của AVR có chiều dài 16 hoặc 32 bit.



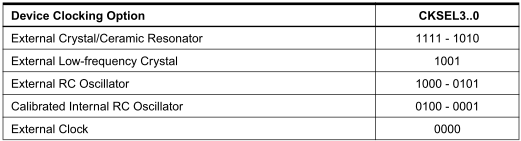
Hình : Bộ nhớ Flash

* + Bộ nhớ SRAM: 2K byte dùng để lưu biến và đánh địa chỉ.

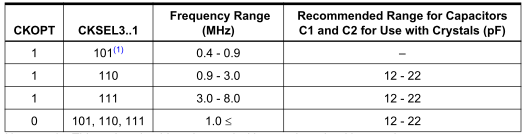


Hình : Bộ nhớ SRAM

* + Bộ nhớ Eeprom: 1K bytes. Bộ nhớ Eeprom không bị mất dữ liệu khi mất nguồn, có thể ghi đọc trong lúc chạy chương trình, dùng để lưu các tham số.
* Nguồn dao động:
  + Bên trong AVR đã có sẵn một bộ giao động có thể lựa chọn với tần số từ 1Mhz – 8 Mhz (mặc định 1Mhz). Do đó nó có thể chạy mà không cần mắc thêm nguồn dao động ngoài. Muốn sử dụng nguồn dao động ngoài chúng ta phải lập trình cho các bít CKSEL3..0 và CKOPT theo bảng:



Hình : Lựa chọn Clock

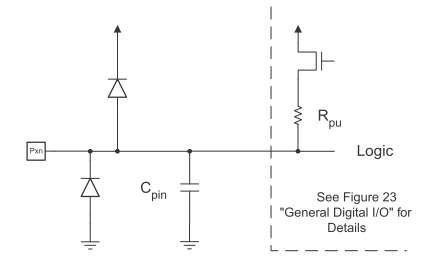


Hình : Lựa chọn tần số hoạt động

* Các ngắt:
  + Khi có ngắt sảy ra, vi điều khiển sẽ tự động lưu các tham số của chương trình và nhảy đến chương trình con thực hiện ngắt.
  + Bảng : Các ngắt của Atmega32

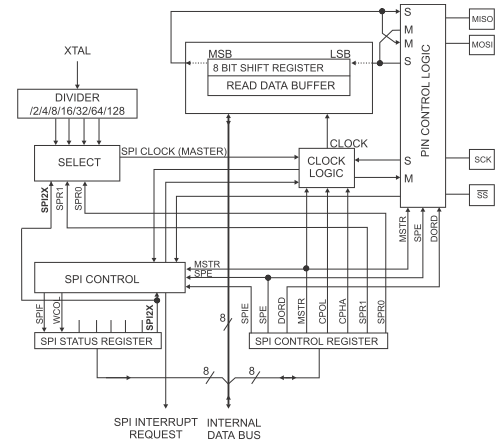


* Các cổng vào ra:
  + Các cổng vào ra và các thanh ghi dùng để điều khiển hướng của cổng được lập trình giá trị tới từng bit. Bộ đệm cổng có thể đủ mạnh để điều khiển hiển thị trực tiếp LED sáng. Tất cả các chân đều có các điện trở kéo lên bên trong.

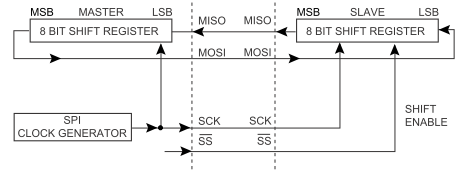


Hình : Sơ đồ nguyên lý tương đương cổng vào ra.

* + Mỗi cổng vào ra có thanh ghi dữ liệu PORTx ( PORTA, PORTB, PORTC, PORTD), thanh ghi hướng dữ liệu DDRx và thanh ghi dữ liệu vào PINx. Khi bit thanh ghi hướng dữ liệu bằng “1” thì chân đó là chân ra, bằng “0” là chân vào.
* Giao tiếp SPI ( Serial Peripheral Interface)
  + Giao tiếp ngoại vi nối tiếp SPI cho phép trao đổi dữ liệu đồng bộ tốc độ cao giữa Atmega32 và các thiết bị ngoại vi hoặc giữa các thiết bị AVR với nhau.
  + . Giao diện ngoại vi nối tiếp (SPI) bao gồm các đặc điểm dưới đây
    - Truyển song công, truyền dữ liệu đồng bộ 3 dây (Three wire)
    - Chế độ hoạt động Master / Slave
    - Chuyển dữ liệu MSB First hoặc LSB First
    - 7 tốc độ bit có thể lập trình
    - Cờ ngắt cuối phiên truyền
    - Cờ ngăn xung đột ghi
    - Đánh thức khỏi chế độ chờ Idle
    - Chế độ SPI Master tốc độ kép (CK/2 )
  + Sơ đồ khối:

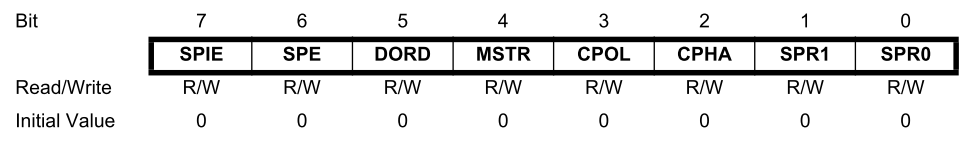


Hình : Sơ đồ khối giao tiếp SPI.

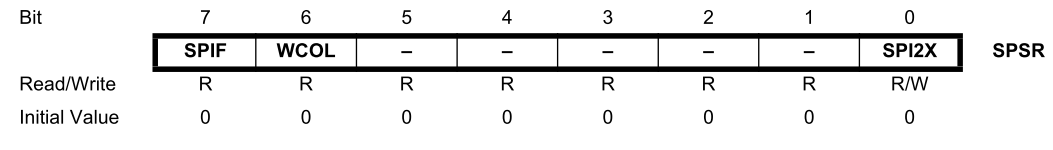


Hình : Kết nối giao tiếp SPI

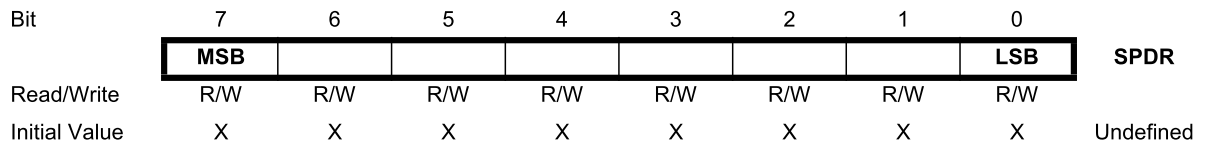
* + Hoạt động:
    - Master sẽ khởi tạo phiên giao tiếp bằng cách kéo chân lựa chọn Slave SS xuống mức thấp. Master và Slave chuẩn bị dữ liệu để gửi vào các thanh ghi dịch tương ứng của chúng , Master tạo ra xung nhịp trên đường SCK để chuyển tiếp dữ liệu . Dữ liệu được chuyển từ Master sang Slave trên đầu ra Master Out - Slave In , MOSI, và từ Slave sang Master bằng chân Master In - Slave Out , MISO. Sau mỗi gói dữ liệu , Master sẽ đồng bộ hóa với Slave bằng một xung cao trên đường lựa chọn Slave SS.
    - Khi được cấu hình là một Master , giao diện SPI không tự động điều khiển 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 việc này được thực hiện ,ghi một 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 một Byte , bộ tạo xung nhịp SPI dừng lại, đặt cờ báo kết thúc quá trình chuyển dữ liệu (SPIF) . Nếu như ngắt SPI được kích hoạt (bít SPIE ) trong thanh ghi SPCR, một ngắt được truy vấn . Master có thể tiếp tục chuyển byte dữ liệu tiếp theo bằng cách ghi vào trong thanh ghi SPDR hoặc tạo tín hiệu kết thúc gói dữ liệu bằng xung cao trên đường lựa chọn Slave, nhánh SS. Byte đến cuối cùng sẽ được giữ trong bộ đệm của thanh ghi cho lần sử dụng sau.
    - Khi được cấu hình là Slave , giao diện SPI sẽ ở chế độ ngủ khi 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 dịch chuyển ra ngoài bằng xung nhịp trên chân SCK cho đến khi chân SS được điều khiển ở mức thấp. Kết thúc truyền một byte nó cũng tạo ra ngắt nếu như được lập trình. Slave có thể tiếp tục đặt dữ liệu mới để gửi vào trong thanh ghi SPDR. Byte đến cuối cùng sẽ được giữ lại sử dụng sau.
  + Thanh ghi điều khiển SPCR



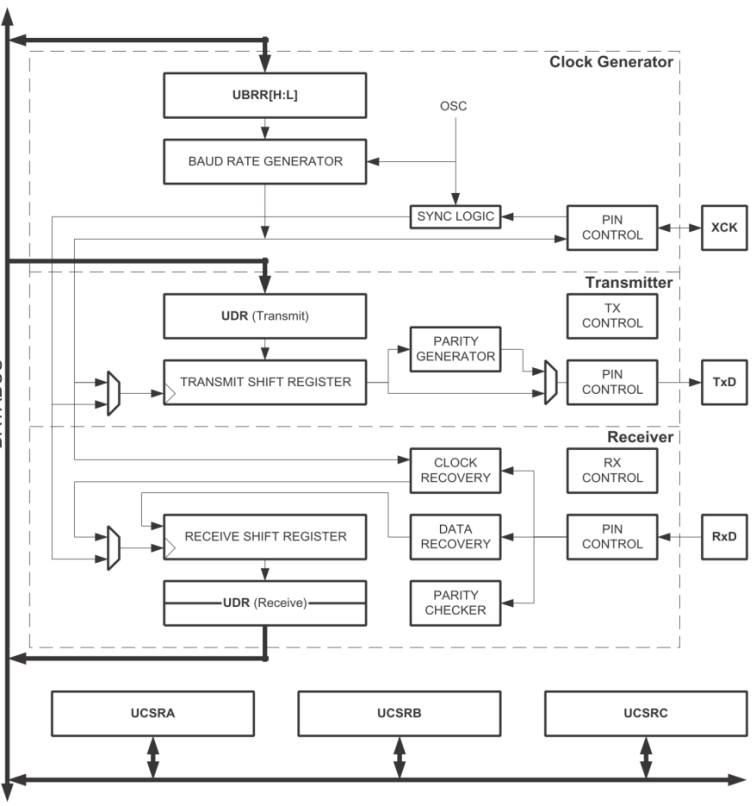
* Bit 7 – SPIE: Cho phép ngắt SPI
* Bit 6 – SPE: Cho phép SPI
* Bit 5 – DORD: Thứ tự dữ liệu. DORD = 1 , LSB first, DORD = 0, MSB first.
* Bit 4 – MSTR: Chọn Master/Slave, 1 Master, 0 Slave.
* Bit 3 – CPOL: Chọn cực Clock, bằng 1 SCK ở mức cao khi rảnh, bằng 0 SCK ở mức thấp khi rảnh.
* Bit 2 – CPHA: Pha Clock
* Bits 1, 0 – SPR1, SPR0: chọn tốc độ đồng hồ.
* Thanh ghi trạng thái SPSR



* Bit 7 – SPIF: Cờ ngắt SPI
* Bit 6 – WCOL: Cờ xung đột ghi
* Bit 5..1 – Res: Bit dự trữ
* Bit 0 – SPI2X: Bit nhân đôi tốc độ.
* Thanh ghi dữ liệu SPDR

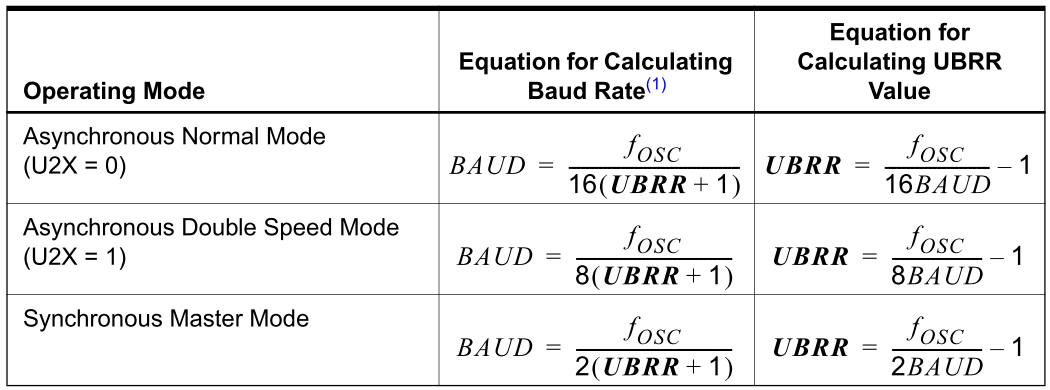


* Giao tiếp USART
  + USART: Universal Synchronous and Asynchronous serial Receiver and Transmitter - Thu phát nối tiếp đồng bộ và không đồng bộ chung là giao tiếp nối tiếp có độ linh hoạt cao. Giao tiếp tại các chân TXD, RXD, XCK của vi điều khiển. Các tính năng chính:
    - Hoạt động song công.
    - Hai chế độ đồng bộ và không đồng bộ.
    - Hoạt động đồng bộ xung Master hay 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.
    - Tạo toàn vẹn dữ liệu chẵn, lẻ và hỗ trợ kiểm tra tính chẵn lẻ bằng phần cứng .
    - Dò tràn dữ liệu.
    - Dò lỗi khung truyền.
    - Bộ lọc nhiễu bao gồm dò tìm bit khởi động sai và bộ lọc số thông thấp.
    - 3 ngắt riêng biệt: Phát xong, trống thanh ghi dữ liệu TX , thu xong.
    - Chế độ truyền thông nhiều bộ sử lý.
    - Chế độ truyền thông không đồng bộ tốc độ kép.
  + Sơ đồ khối:



Hình : Sơ đồ khối bộ USART

* + Tính toán tốc độ Baud:
    - Bảng : Tính toán tốc độ baud:



Baud: Tốc độ baud ( bps)

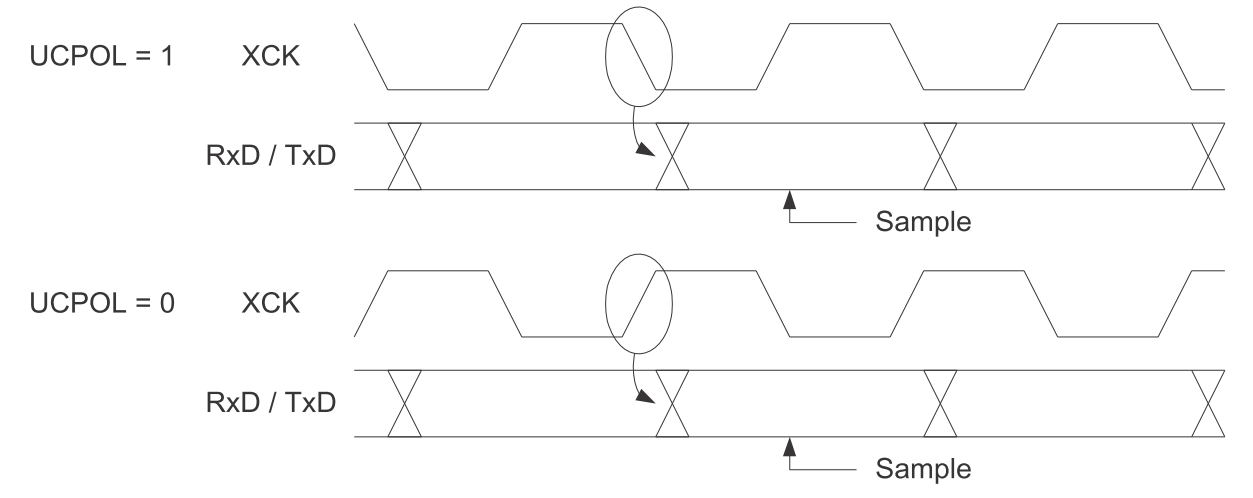
Fosc: Tần số dao động đồng hồ hệ thống

UBRR: Nội dung của thanh ghi UBRRH và UBRRL, ( 0 – 4095)

* + Tốc độ nhân đôi:

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 chế độ không đồng bộ. Cài đặt bit này là 0 khi sử dụng chế độ điều khiển đồng bộ.

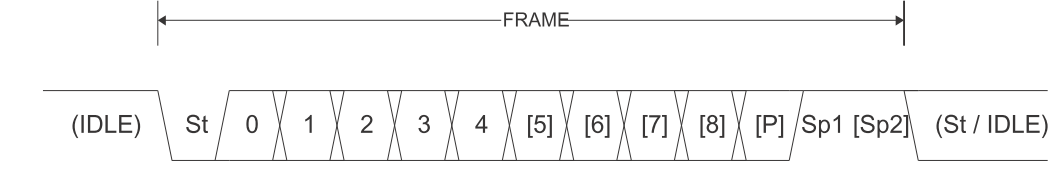
* + Định thời chế độ đồng bộ:



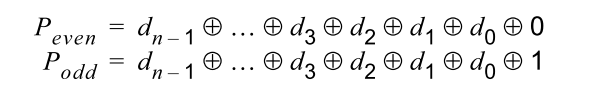
Các bit UCPOL và Bit UCRSC lựa chọn sườn xung XCK được sử dụng

cho việc lấy mẫu dữ liệu và thay đổi dữ liệu. khi UCPOL = 0 thì dữ liệu sẽ thay đổi tại sườn lên của xung XCK và được lấy mẫu tại sườn xuống của xung XCK . Nếu UCPOL = 1 , dữ liệu sẽ thay đổi tại sườn xuống của xung XCK và được lấy mẫu tại sườn lên của xung XCK.

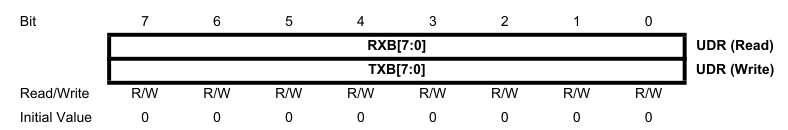
* + Các dạng khung dữ liệu – Frame Formats
    - Một khung được xác định bằng một ký tự gồm các bit dữ liệu, các bit bộ đồng bộ hóa (các bit start và stop ), và bit chẵn lẻ cho việc kiểm tra các lỗi . USART chấp nhận tất cả 30 kết hợp của các định dạng frame sau:
    - 1 bit start
    - 5 ,6 ,7 ,8 hoặc 9 bit dữ liệu
    - Không có , hoặc có các bit chẵn lẻ
    - 1 hoặc 2 bit stop



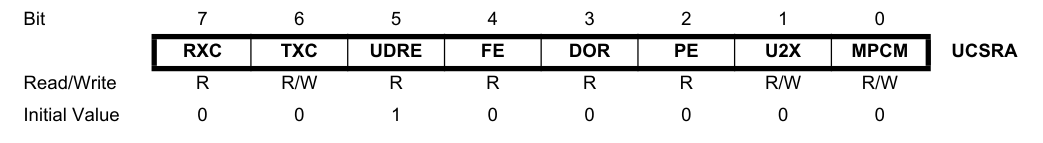
Hình : Định dạng khung dữ liệu.

* + - 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
  + Tính toán bit chẵn lẻ - Parity Bit Calculation
    - Bit chẵn lẻ được tính toán bằng cách cộng module tất cả các bit dữ liệu . Nếu bit lẻ được sử dụng, kết quả của quá trình cộng module được đảo ngược . Sự liên quan giữa bit chẵn lẻ và các bit dữ liệu như bên dưới 

* + - P even : bit chẵn lẻ sử dụng bậc chẵn
    - P 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 .
  + Các thanh ghi USART:
    - Thanh ghi dữ liệu I/O



* + - Thanh ghi điều khiển và trạng thái USART A – UCSRA



Bit7 – RCX: hoàn thành thu nhận tín hiệu

Bit cờ này được đặt khi có các dữ liệu không được đọc trong bộ đệm nhận và bị xóa khi bộ đệm nhận trống

Bit 6 – TXC : hoàn thành chuyển dữ liệu USART

Bit 5 – UDRE : báo trống thanh ghi dữ liệu USART

Bit 4 – Fe : lỗi khung truyền

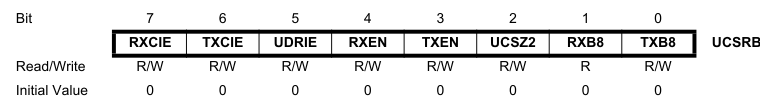
Bit 3 – DOR : báo tràn dữ liệu

Bit 2 – UPE : lỗi chẵn lẻ

Bit 1 – U2X: tốc độ truyền dữ liệu USART kép

Bit 0 – MPCM : chế độ truyền thông đa xử lý

* + - Thanh ghi trạng thái và điều khiển USART B – UCSRB



Bit 7 – RXCIE : kích hoạt ngắt hoàn thành RX

Bit 6 – TXCIE : kích hoạt ngắt hoàn thành TX

Bit 5 – UDRIE : kích hoạt ngắt trống thanh ghi dữ liệu USART

Bit 4 – RXEN : kích hoạt bộ thu tín hiệu

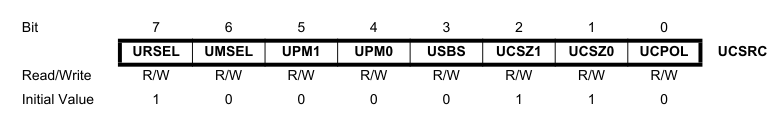
Bit 3 – TXEN : kích hoạt bộ chuyển phát

Bit 2 – UCSZ2 : kích cỡ chuỗi kí tự

Bit 1 – RXB8 : bit 8 dữ liệu đến

Bit 0 – TXB8 : bit 8 dữ liệu chuyển phát

* + - Thanh ghi trạng thái và điều khiển USART C – UCSRC



Bit 7 – bit dự trữ

Bit 6 – UMSELn : lựa chọn chế độ USART

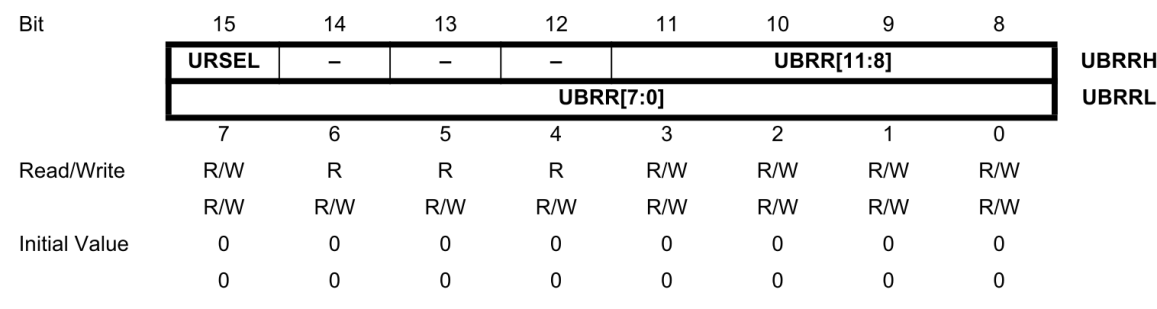
Bit 5:4 – UPMn1:0 : chế độ chẵn lẻ

Bit 3 – USBSn : lựa chọn bit stop

Bit 2:1 – UCSZn1:0 : kích cỡ chuỗi kí tự

Bit 0 – UCPOLn : cực xung nhịp

* + - Các thanh ghi Baud Rate USART – UBRRL và UBRRH



Bit 15: URSEL: Bit lựa chọn thanh ghi UBRRH hay UCSRC.

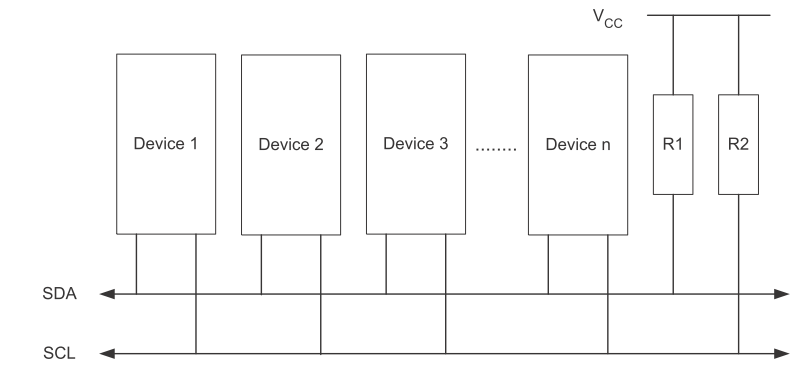
Bit 14:12: Bỉt dự trữ.

Bit 11:0 – UBRR11:0 : thanh ghi baud rate USART

* Giao tiếp hai dây tuần tự \_ Two – wire Serial Inteface ( TWI, I2C)
  + Đặc điểm
    - Đơn giản nhưng mạnh mẽ và 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
    - Mạch loại bỏ nhiễu loại bỏ đỉnh nhọn trên các đường bus
    - Đầy đủ các địa chỉ slave lập trình được với sự hỗ trợ gọi chung
    - Khi nhận ra địa, AVR bị đáng thức trong các chế độ sleepmode.

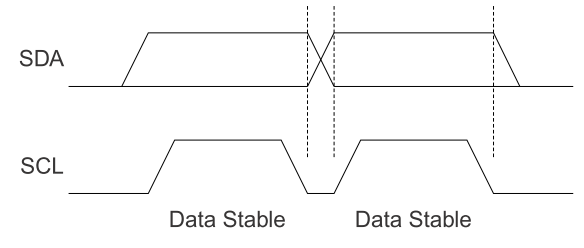
* + Định nghĩa bus giao diện tuần tự hai dây

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) . Phần cứng bên ngoài để điều khiển bus chỉ cần 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 cơ chế cho việc giải quyết nội dung bus đã sẵn có trong giao thức TWI.

 Hình : Kết nối TWI bus

* + Trao đổi dữ liệu và định dạng khung:
    - Quá trình chuyển các bít

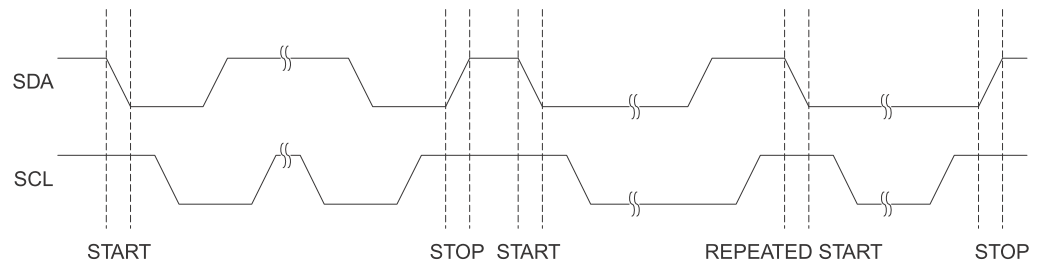
Mỗi bít dữ liệu trên bus TWI được kèm theo một xung trên đường truyền xung nhịp. Mức của đường dữ liệu phải ổn định khi đườn xung nhịp ở mức cao.



Hinh : Trao đổi dữ liệu TWI

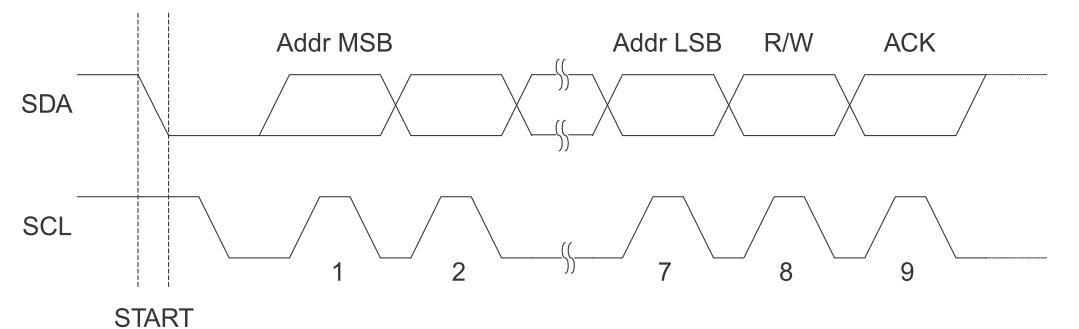
* + - Các điều kiện khởi động và dừng truyền:

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 Master nào khác có thể can thiệp được. Như giản 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.



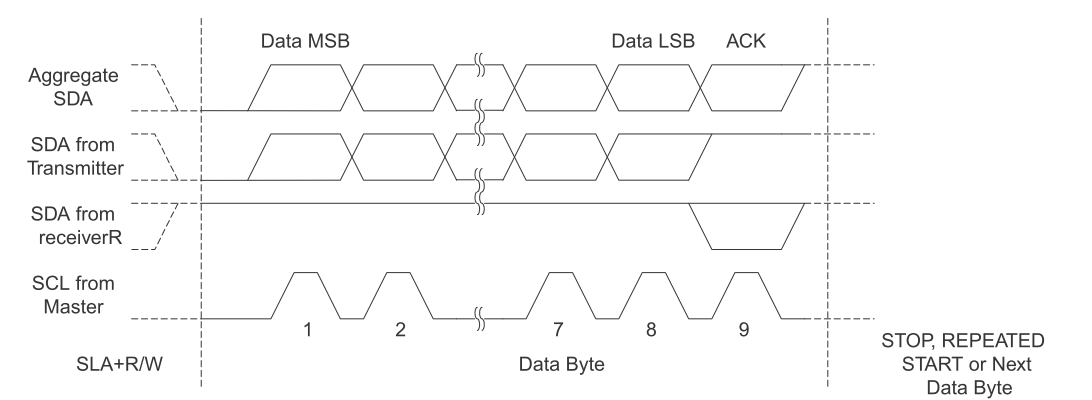
Hình : Điều kiện start và stop TWI

* + Định dạng khung địa chỉ:



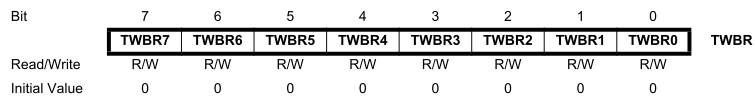
Hình : Định dạng khung địa chỉ TWI

* + Định dạng gói dữ liệu:



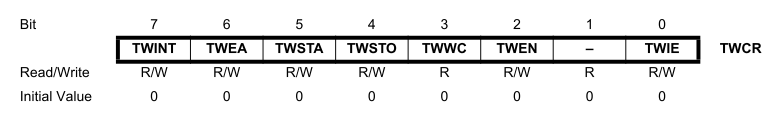
Hình : Định dạng gói dữ liệu.

* + Các thanh ghi TWI
    - Thanh ghi tốc độ bit TWI – TWBR



Bit 7..0 thanh ghi tốc độ bit TWI

* Thanh ghi điều khiển TWI – TWCR



Bit 7 – TWINT : cờ ngắt TWI

Bit 6 – TWEA : bit nhận biết kích hoạt TWI

Bit 5 – TWSTA : bit điều kiện khởi động TWI

Bit 4 – TWSTO : bit điều kiện STOP TWI

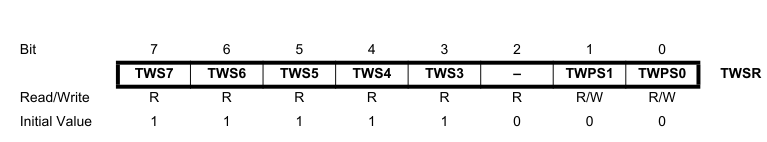
Bit 3 – TWWC – cờ báo viết xung đột viết TWI

Bit 2 – TWEN : bit kích hoạt TWI

Bit 1 – Res : bit dự trữ

Bit 0 – TWIE : kích hoạt ngắt TWI

* Thanh ghi trạng thái TWI – TWSR

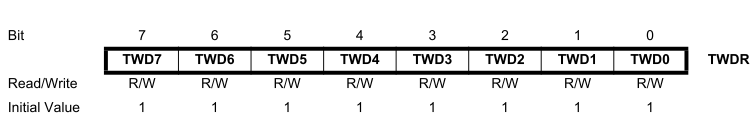


Bit 7..3 – TWS : trạng thái TWI

Bit 2 – Res : bit dự trữ

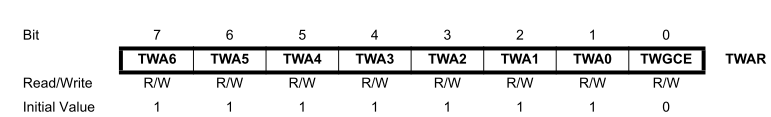
Bit 1..0 – TWPS : các bit đếm gộp trước TWI

* Thanh ghi dữ liệu TWI – TWDR



Bit 7...0 – TWD : thanh ghi dữ liệu TWI

* Thanh ghi địa chỉ TWI (Slave ) – TWAR



Bit 7..1 –TWA : thanh ghi địa chỉ TWI (slave )

Bit 0 – TWGCE : bit kích hoạt nhận diện gọi chung TWI