

TECNICATURA SUPERIOR EN TELECOMUNICACIONES

ELECTRÓNICA MICROCONTROLADA

Docentes: Ing. Jorge E. Morales, Téc. Gonzalo Vera.

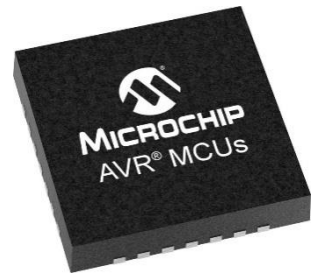
Título: Estructura del Microcontrolador AVR® de 8 bits

Grupo 1:

- ❖ Birge, Adolfo Federico.
- ❖ Carunchio, Carlos Javier.
- ❖ Ferreyra, María Luciana.
- ❖ Gutiérrez, Emma Vilma.
- ❖ Merlo, Emmanuel.
- ❖ Romero, Gisela de Lourdes.

Estructura del Microcontrolador AVR® de 8 bits

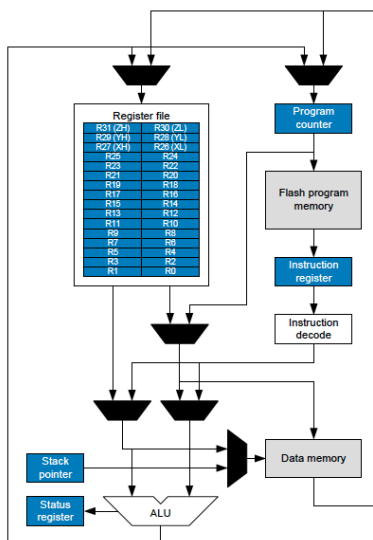
Los microcontroladores (MCU) AVR de 8 bits ofrecen una familia de dispositivos únicos y versátiles. Con la gran cantidad de periféricos incorporados, siempre es útil tener capacitación sobre cómo funcionan, cómo configurarlos y ejemplos de cómo usarlos. A continuación, detallaremos su arquitectura:



Núcleo de CPU AVR

La función principal del núcleo de la Unidad Central de Procesamiento (CPU) AVR ® es garantizar la correcta ejecución del programa. Por lo tanto, la CPU debe poder acceder a las memorias, realizar cálculos, controlar periféricos y manejar interrupciones.

Diagrama de bloques de la arquitectura AVR



Núcleo AVR

Para maximizar el rendimiento y el paralelismo, el AVR utiliza una arquitectura Harvard con memorias y buses separados para programas y datos. Las instrucciones en la memoria del programa se ejecutan con canalización de un solo nivel. Mientras se ejecuta una instrucción, la siguiente instrucción se obtiene previamente de la memoria del programa. Este concepto permite ejecutar instrucciones en cada ciclo de reloj. La memoria del programa es una memoria Flash reprogramable en el sistema.

Registros

El archivo de registro de acceso rápido contiene 32 registros de trabajo de propósito general de 8 bits con un solo tiempo de acceso de ciclo de reloj. Seis de los 32 registros se pueden utilizar como tres punteros de registro de direcciones indirectas de 16 bits para el direccionamiento del espacio de datos, lo que

permite cálculos de direcciones eficientes. Uno de estos punteros de dirección también se puede utilizar como puntero de dirección para tablas de búsqueda en la memoria de programa Flash. Estos registros de funciones adicionales son los registros X, Y y Z de 16 bits.

Unidad Aritmética Lógica (ALU)

La ALU admite operaciones aritméticas y lógicas entre registros o entre una constante y un registro. El tiempo de acceso de ciclo de reloj único permite operaciones de ALU de ciclo único. En una operación ALU típica, se emiten dos operandos desde el archivo de registro, se ejecuta la operación y el resultado se almacena nuevamente en el archivo de registro en un ciclo de reloj. Las operaciones de registro único también se pueden ejecutar en la ALU. Después de una operación aritmética, el registro de estado se actualiza para reflejar información sobre el resultado de la operación. El flujo del programa lo proporcionan las instrucciones de salto y llamada condicionales e incondicionales, capaces de abordar directamente todo el espacio de direcciones. La mayoría de las instrucciones AVR tienen un solo formato de palabra de 16 bits. Cada dirección de memoria de programa contiene una instrucción de 16 o 32 bits.

Memoria

Los espacios de memoria en la arquitectura AVR son todos mapas de memoria lineales y regulares. El espacio de la memoria flash del programa se divide en dos secciones, la sección del programa de arranque y la sección del programa de aplicación. Ambas secciones tienen bits de bloqueo dedicados para protección contra escritura y lectura/escritura. La instrucción Store Program Memory (SPM) que escribe en la sección de memoria Flash de la aplicación debe residir en la sección Boot Program.

Durante las interrupciones y las llamadas a subrutinas, el Contador de programa (PC) de la dirección de retorno se almacena en la pila. La pila se asigna efectivamente en la SRAM de datos generales y, en consecuencia, el tamaño de la pila solo está limitado por el tamaño total de la SRAM y el uso de la SRAM.

Todos los programas de usuario deben inicializar el puntero de pila (SP) en la rutina de reinicio (antes de que se ejecuten las subrutinas o interrupciones). El SP es accesible para lectura/escritura en el espacio de E/S. Se puede acceder fácilmente a la SRAM de datos a través de los cinco modos de direccionamiento diferentes admitidos en la arquitectura AVR.

El espacio de memoria de E/S contiene 64 direcciones para funciones periféricas de la CPU como registros de control, interfaz periférica en serie (SPI) y otras funciones de E/S. Se puede acceder a la memoria de E/S directamente o como ubicaciones de espacio de datos que siguen a las del archivo de registro, 0x20 - 0x5F. Además, este dispositivo ha ampliado el espacio de E/S de 0x60 - 0xFF en SRAM.

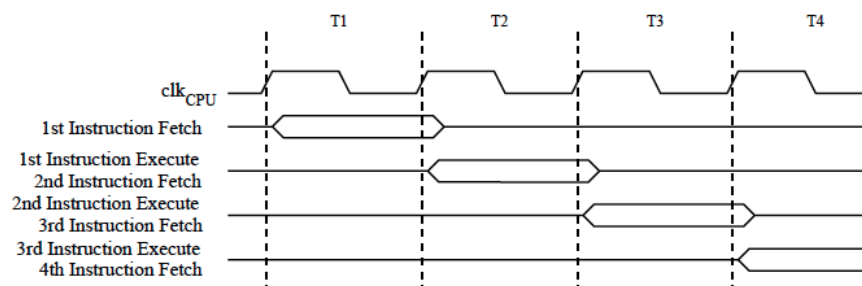
Interrupciones

Un módulo de interrupción flexible tiene sus registros de control en el espacio de E/S con un bit de habilitación de interrupción global adicional en el registro de estado. Todas las interrupciones tienen un vector de interrupción separado en la tabla de vectores de interrupción. Las interrupciones tienen prioridad de acuerdo con su posición en el vector de interrupción. Cuanto menor sea la dirección del vector de interrupción, mayor será la prioridad.

Tiempo de instrucción AVR

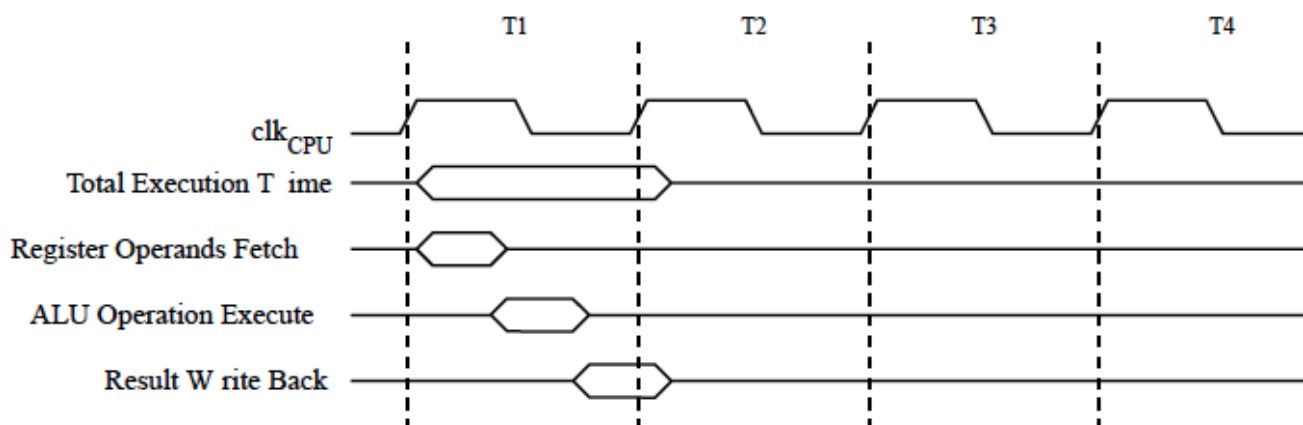
La unidad central de procesamiento (CPU) AVR es impulsada por el reloj de la `clkCPU` CPU, generado directamente desde la fuente de reloj seleccionada para el chip. No se utiliza ninguna división de reloj interna. La arquitectura de Harvard y el concepto de archivo de registro de acceso rápido permiten obtener y ejecutar instrucciones en paralelo. Este es el concepto básico de canalización para obtener hasta 1 MIPS por MHz con los resultados únicos correspondientes para funciones por costo, funciones por relojes y funciones por unidad de potencia.

Obtención de instrucciones en paralelo y ejecución de instrucciones



En un solo ciclo de reloj, se ejecuta una operación de unidad lógica aritmética (ALU) utilizando dos operandos de registro y el resultado se almacena nuevamente en el registro de destino.

Operación ALU de ciclo único



Unidad lógica aritmética AVR

La Unidad Aritmética Lógica (ALU) AVR® de alto rendimiento opera en conexión directa con los 32 registros de trabajo de propósito general. Dentro de un solo ciclo de reloj, se ejecutan operaciones aritméticas entre registros de Propósito General o entre un registro y un inmediato. Las operaciones ALU se dividen en tres categorías principales: funciones aritméticas, lógicas y de bits. Algunas implementaciones de la arquitectura también proporcionan un poderoso multiplicador que admite tanto la multiplicación con signo/sin signo como el formato fraccionario.

ARITHMETIC AND LOGIC INSTRUCTIONS					
Mnemonics	Operands	Description	Operation	Flags	#Clocks
ADD	Rd, Rr	Add two Registers without Carry	$Rd \leftarrow Rd + Rr$	Z,C,N,V,H	1
ADC	Rd, Rr	Add two Registers with Carry	$Rd \leftarrow Rd + Rr + C$	Z,C,N,V,H	1
SUB	Rd, Rr	Subtract two Registers	$Rd \leftarrow Rd - Rr$	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	$Rd \leftarrow Rd - K$	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract two Registers with Carry	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,H	1
SBCI	Rd, K	Subtract Constant from Reg with Carry.	$Rd \leftarrow Rd - K - C$	Z,C,N,V,H	1
AND	Rd, Rr	Logical AND Registers	$Rd \leftarrow Rd \cdot Rr$	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	$Rd \leftarrow Rd \cdot K$	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	$Rd \leftarrow Rd \vee Rr$	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	$Rd \leftarrow Rd \vee K$	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	$Rd \leftarrow Rd \oplus Rr$	Z,N,V	1
COM	Rd	One's Complement	$Rd \leftarrow 0xFF - Rd$	Z,C,N,V	1
NEG	Rd	Two's Complement	$Rd \leftarrow 0x00 - Rd$	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	$Rd \leftarrow Rd \cdot (0xFF - K)$	Z,N,V	1
INC	Rd	Increment	$Rd \leftarrow Rd + 1$	Z,N,V	1
DEC	Rd	Decrement	$Rd \leftarrow Rd - 1$	Z,N,V	1
TST	Rd	Test for Zero or Minus	$Rd \leftarrow Rd \cdot Rd$	Z,N,V	1
CLR	Rd	Clear Register	$Rd \leftarrow Rd \oplus Rd$	Z,N,V	1
SER	Rd	Set Register	$Rd \leftarrow 0xFF$	None	1

Registros de propósito general AVR

La estructura del archivo de registro AVR® está optimizada para el conjunto de instrucciones de la computadora del conjunto de instrucciones reducido mejorado (RISC) del AVR. Para lograr el rendimiento y la flexibilidad requeridos, el archivo de registro admite los siguientes esquemas de E/S:

Un operando de salida de 8 bits y una entrada de resultado de 8 bits.

Dos operandos de salida de 8 bits y una entrada de resultado de 8 bits.

Dos operandos de salida de 8 bits y una entrada de resultado de 16 bits.

Un operando de salida de 16 bits y una entrada de resultado de 16 bits.

Conjunto de instrucciones ALU

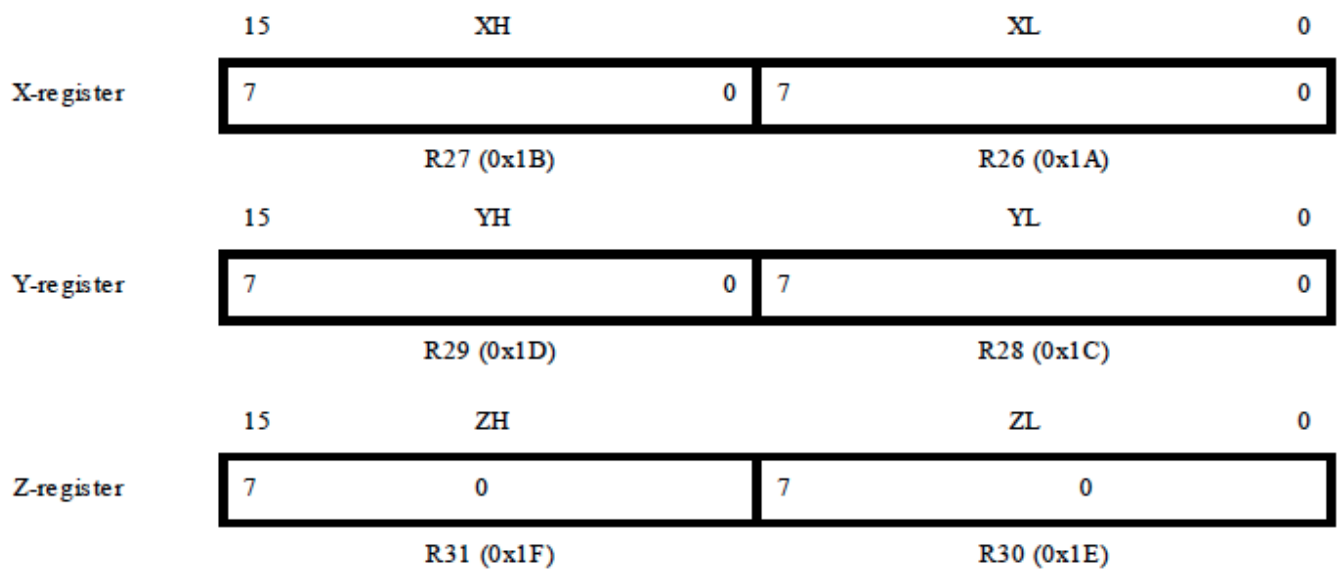
Registros de trabajo de propósito general de la CPU AVR

	7	0	Addr.	
General Purpose Working Registers	R0		0x00	
	R1		0x01	
	R2		0x02	
	...			
	R13		0x0D	
	R14		0x0E	
	R15		0x0F	
	R16		0x10	
	R17		0x11	
	...			
	R26		0x1A	X-register Low Byte
	R27		0x1B	X-register High Byte
	R28		0x1C	Y-register Low Byte
	R29		0x1D	Y-register High Byte
	R30		0x1E	Z-register Low Byte
	R31		0x1F	Z-register High Byte

La mayoría de las instrucciones que operan en el archivo de registro tienen acceso directo a todos los registros y la mayoría de ellas son instrucciones de un solo ciclo. A cada registro también se le asigna una dirección de memoria de datos, asignándolos directamente a las primeras 32 ubicaciones del espacio de datos del usuario. Aunque no se implementa físicamente como ubicaciones SRAM, esta organización de memoria proporciona una gran flexibilidad en el acceso a los registros, ya que los registros de puntero X, Y y Z se pueden configurar para indexar cualquier registro en el archivo.

El registro X, el registro Y y el registro Z

Los registros R26 a R31 tienen algunas funciones adicionales a su uso de propósito general. Estos registros son punteros de dirección de 16 bits para el direccionamiento indirecto del espacio de datos. Los tres registros de direcciones indirectas (X, Y y Z) se definen como se describe en la figura.



Registro de pila AVR

La pila se utiliza principalmente para almacenar datos temporales, variables locales y direcciones de retorno después de interrupciones y llamadas a subrutinas. Se implementa como un crecimiento de ubicaciones de memoria superiores a inferiores. El registro del puntero de pila siempre apunta a la parte superior de la pila; apunta al área de la pila SRAM de datos donde se encuentran las pilas de subrutinas e interrupciones.

Puntero de pila

Bit	15	14	13	12	11	10	9	8	
0x3E	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
0x3D	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	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	
	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	

El AVR Stack Pointer se implementa como dos registros de 8 bits en el espacio de E/S. El número de bits realmente utilizados depende de la implementación.

Conjunto de instrucciones de pila

Instruction	Stack pointer	Description
PUSH	Decrementated by 1	Data is pushed onto the stack
ICALL RCALL	Decrementated by 2	Return address is pushed onto the stack with a subroutine call or interrupt
POP	Incrementated by 1	Data is popped from the stack
RET RETI	Incrementated by 2	Return address is popped from the stack with return from subroutine or return from interrupt

Un comando PUSH de pila disminuirá el puntero de pila. El programa debe definir la pila en la SRAM de datos antes de que se ejecuten las llamadas a subrutinas o se habiliten las interrupciones. El valor inicial del Puntero de pila es igual a la última dirección de la SRAM interna y el Puntero de pila debe configurarse para que apunte por encima del inicio de la SRAM.

Registro de estado de AVR

El registro de estado contiene información sobre el resultado de la última instrucción aritmética ejecutada. Esta información se puede utilizar para alterar el flujo del programa con el fin de realizar operaciones condicionales. El registro de estado se actualiza después de todas las operaciones de la unidad lógica aritmética (ALU). En muchos casos, esto eliminará la necesidad de usar las instrucciones de comparación dedicadas, lo que dará como resultado un código más rápido y compacto.

ESTADO: Registro de estado

L/E-0/0	L/E-0/0	L/E-0/0	L/E-0/0	L/E-0/0	L/E-0/0	L/E-0/0	L/E-0/0
yo	T	H	S	V	norte	Z	C
bit 7				bit 0			

bit 7

I: Habilitación de interrupción global: el bit de habilitación de interrupción global debe establecerse para que se habiliten las interrupciones. El control de habilitación de interrupción individual se realiza entonces en registros de control separados. Si se borra el registro de Habilitación de interrupción global, ninguna de las interrupciones se habilita independientemente de la configuración de habilitación de interrupción individual. El bit I (bit 7) se borra por hardware

después de que se ha producido una interrupción y se establece mediante la instrucción RETI (Return from Interrupt) para habilitar las interrupciones subsiguientes. La aplicación también puede establecer y borrar el bit I con las instrucciones Set Global Interrupt Flag (SEI) y Clear Global Interrupt Flag (CLI), como se describe en la referencia del conjunto de instrucciones.

bit 6

T: almacenamiento de copia: las instrucciones de copia de bits, carga de bits (BLD) y almacenamiento de bits (BST), utilizan el bit T como fuente o destino para el bit operado. La instrucción BST puede copiar un bit de un registro en el archivo de registro en T y la instrucción BLD puede copiar un bit en T en un bit en un registro en el archivo de registro.

bit 5

H: Indicador de medio acarreo: El indicador de medio acarreo, H, indica medio acarreo en algunas operaciones aritméticas. Es útil en la aritmética decimal de código binario (BCD).

bit 4

S: Bandera de signo, $S = N \text{ xor } V$: El bit S es siempre un exclusivo o entre la Bandera negativa y la Bandera de desbordamiento del complemento a dos.

bit 3

V: Indicador de desbordamiento de complemento a dos: El indicador de desbordamiento de complemento a dos, V, es compatible con la aritmética de complemento a dos.

bit 2

N: Bandera Negativa: La Bandera Negativa, N, indica un resultado negativo en una operación aritmética o lógica.

bit 1

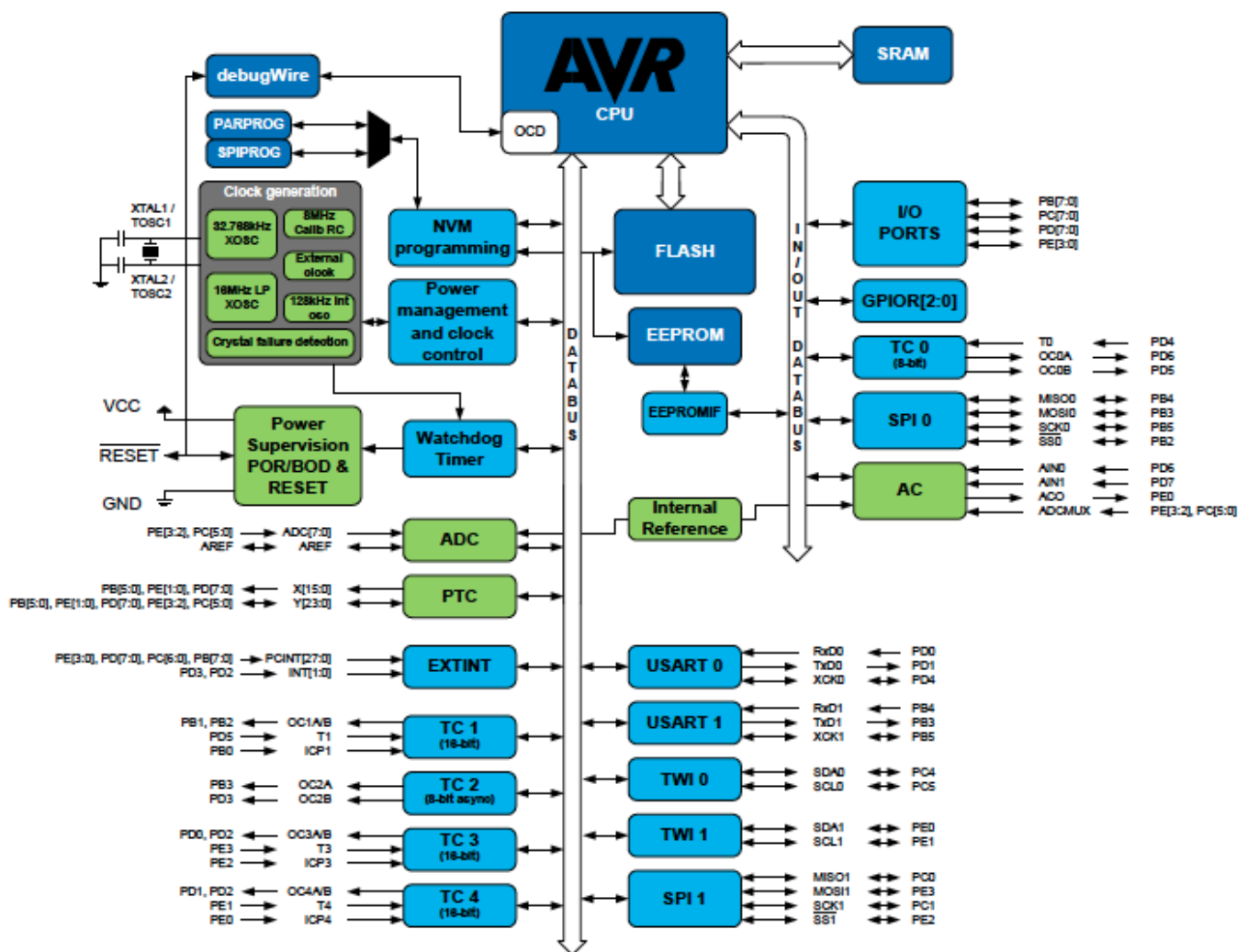
Z: Indicador cero: El indicador cero, Z, indica un resultado cero en una operación aritmética o lógica.

bit 0

C: Indicador de acarreo: El indicador de acarreo, C, indica un acarreo en una operación aritmética o lógica.

Memoria AVR

Los microcontroladores AVR se construyen utilizando una arquitectura Harvard modificada. Esto significa que el espacio de la memoria de programa flash está en un bus de direcciones separado de la memoria estática de acceso aleatorio (SRAM). Hay dos buses de datos, uno que puede acceder a todos los datos y el bus de datos de entrada/salida con acceso limitado a una pequeña sección de memoria.

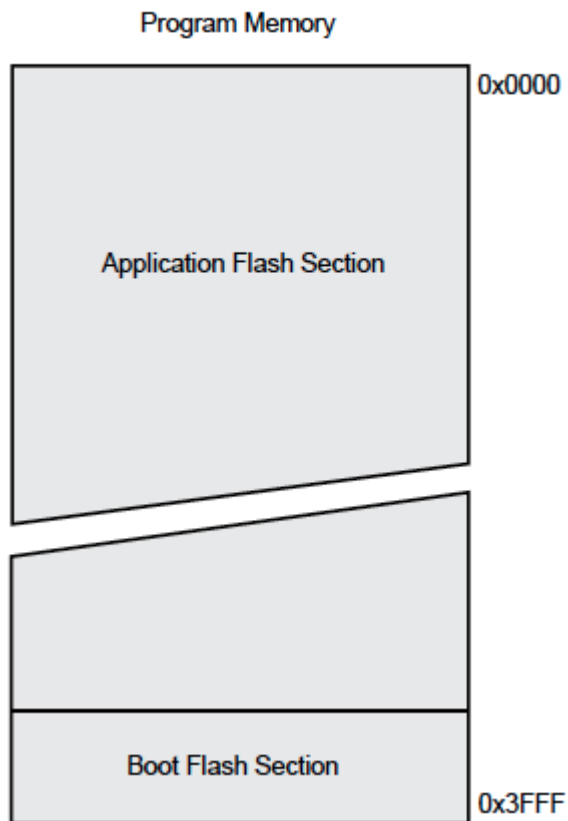


Memoria de programa

Los microcontroladores AVR contienen memoria flash reprogramable en el sistema en chip para el almacenamiento de programas. Dado que todas las instrucciones AVR tienen 16 o 32 bits de ancho, el Flash está organizado como 32K x 16. Para la seguridad del software, el espacio de la memoria del programa Flash se divide en dos secciones: la sección del cargador de arranque y la sección del programa de aplicación en el dispositivo. La memoria Flash tiene una resistencia típica de al menos 10.000 ciclos de escritura/borrado. Las tablas constantes se pueden asignar

dentro de todo el espacio de direcciones de la memoria del programa, utilizando la instrucción Cargar memoria del programa (LPM). También hay una biblioteca de funciones para hacer esto más fácil Biblioteca AVR Libc.

ATmega324PB



Memoria de datos EEPROM

La memoria de solo lectura programable borrable eléctricamente (EEPROM) de datos se organiza como un espacio de datos separado, en el que se pueden leer y escribir bytes individuales. El acceso desde la CPU a la EEPROM se realiza a través de los registros de direcciones de la EEPROM, el registro de datos de la EEPROM y el registro de control de la EEPROM. La EEPROM tiene una resistencia de al menos 100.000 ciclos de escritura/borrado.

Memoria de datos SRAM

Se puede acceder a los datos a través del bus de datos estándar. Hay un bus de entrada/salida secundario para acceso directo rápido a ubicaciones selectas. La memoria de datos consta de:

- Registros
- Memoria de E/S
- Memoria de E/S extendida (depende del dispositivo)

- SRAM interna

Espacio de registro: consta de 32 registros de trabajo de 8 bits de uso general (R0-R31).

Memoria de E/S: contiene espacio direccionable para funciones periféricas, como registros de control y otras funciones de E/S.

Memoria de E/S extendida: algunos microcontroladores AVR con más periféricos necesitan más espacio del que puede ocupar la memoria de E/S, por lo que parte de la SRAM se usa como memoria de E/S extendida para manejar los registros de control de periféricos adicionales y otras funciones de E/S.

SRAM interna (memoria de datos): se utiliza para almacenar temporalmente variables y resultados intermedios dentro de una aplicación de software.

Data Memory	
32 Registers	0x0000 - 0x001F
64 I/O Registers	0x0020 - 0x005F
160 Ext I/O Reg.	0x0060 - 0x00FF
Internal SRAM (2048 x 8)	0x0100
	0x08FF

Hay cinco modos de direccionamiento de bus de datos diferentes (no de entrada/salida) para la memoria de datos:

Directo: el direccionamiento directo alcanza todo el espacio de datos.

Indirecto: en el archivo de registro, los registros R26 a R31 presentan los registros de puntero de direccionamiento indirecto.

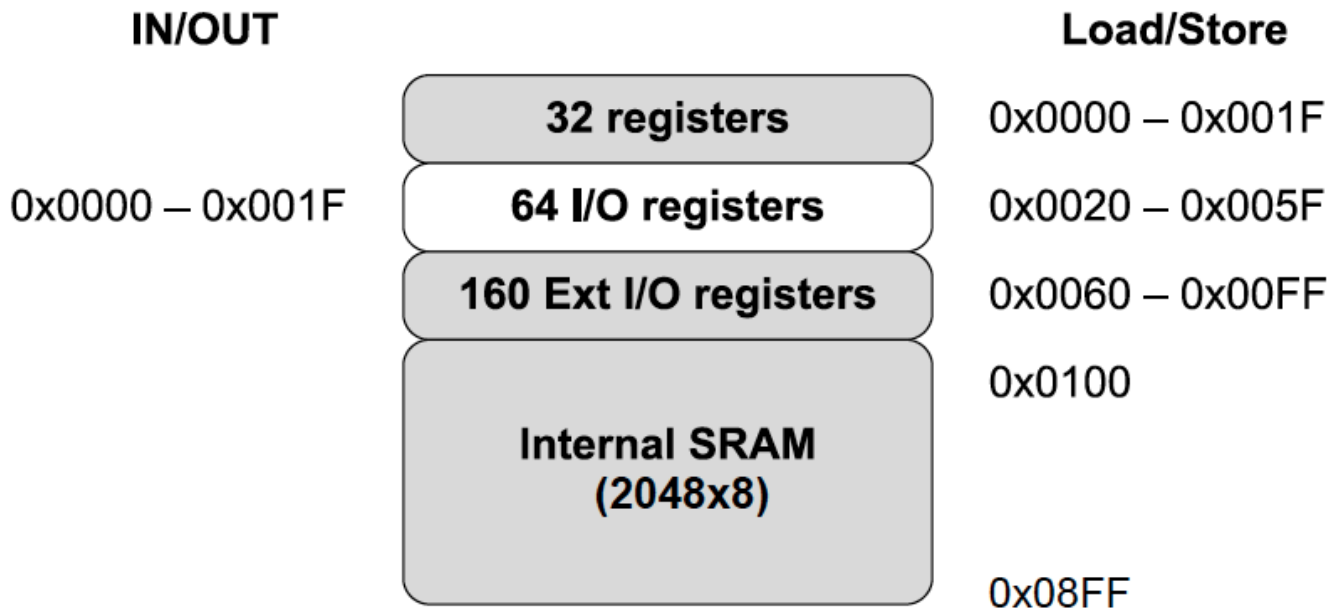
Indirecto con desplazamiento: el modo Indirecto con desplazamiento llega a 63 ubicaciones de direcciones desde la dirección base proporcionada por el registro Y o Z.

Indirecto con decremento previo: los registros de dirección X, Y y Z se reducen.

Indirecto con posincremento: los registros de dirección X, Y y Z se incrementan.

Bus de datos de entrada/salida

Este bus de datos tiene acceso directo a la sección de memoria de E/S de 64 bytes (no extendida) mediante una dirección de 0x00 a 0x1F. También se puede acceder a esta memoria mediante el bus de datos estándar utilizando un desplazamiento de dirección 0x20 en el comando de acceso.



Memoria de E/S

Se puede acceder a todas las ubicaciones de E/S (memoria de E/S y memoria de E/S extendida) mediante las instrucciones de ensamblaje LD/LDS/LDD y ST/STS/STD utilizando el bus de datos estándar. Los datos se transfieren entre los 32 registros de trabajo de propósito general y el espacio de E/S.

Los registros de E/S dentro del rango de direcciones del bus de datos de entrada/salida 0x00-0x1F (memoria de E/S) son accesibles directamente mediante bits mediante las instrucciones SBI y CBI. En estos registros, el valor de los bits individuales se puede comprobar mediante las instrucciones SBIS y SBIC.

Registros de E/S de uso general

Tres registros de E/S de uso general, el registro de E/S de uso general 0/1/2 (GPOR 0/1/2) se encuentran en la parte superior de la memoria de E/S (0x020-0x022). Estos registros se pueden utilizar para almacenar cualquier información y son particularmente útiles para almacenar variables globales y banderas de estado. Se puede acceder directamente a estos registros

mediante las instrucciones SBI, CBI, SBIS y SBIC a través del bus de datos de entrada/salida. Los registros de E/S restantes comienzan después de los registros de E/S de propósito general.

Acceso a registros de 16 bits

El bus de datos AVR tiene 8 bits de ancho, por lo que acceder a registros de 16 bits requiere operaciones atómicas. Se debe acceder a estos registros mediante bytes mediante dos operaciones de lectura o escritura. Los registros de 16 bits están conectados al bus de 8 bits y un registro temporal usando un bus de 16 bits.

Para una operación de escritura, el byte alto del registro de 16 bits debe escribirse antes que el byte bajo. A continuación, el byte alto se escribe en el registro temporal. Cuando se escribe el byte bajo del registro de 16 bits, el registro temporal se copia en el byte alto del registro de 16 bits en el mismo ciclo de reloj.

Para una operación de lectura, el byte bajo del registro de 16 bits debe leerse antes que el byte alto. Cuando la CPU lee el registro de byte bajo, el byte alto del registro de 16 bits se copia en el registro temporal en el mismo ciclo de reloj que se lee el byte bajo. Cuando se lee el byte alto, se lee del registro temporal.

Esto asegura que siempre se acceda simultáneamente a los bytes alto y bajo de los registros de 16 bits al leer o escribir el registro.

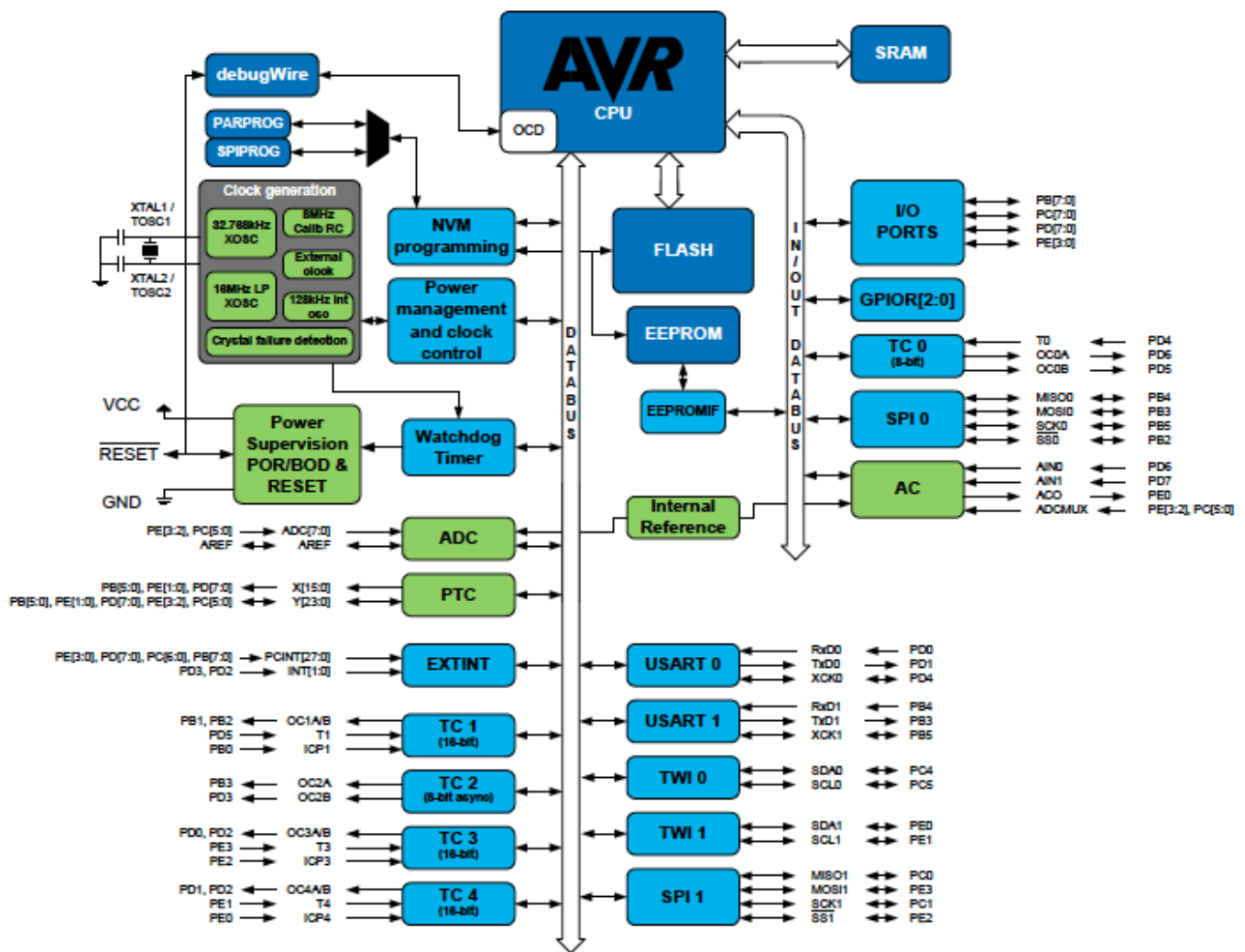
Optimización del código C en AVR

Antes de optimizar el software de los sistemas integrados, es necesario comprender bien cómo está estructurado el núcleo del microcontrolador (MCU) AVR ® y qué estrategias utiliza AVR GNU Compiler Collection (GCC) para generar código eficiente para el procesador.

Arquitectura Atmel AVR de 8 bits

AVR utiliza la arquitectura Harvard, con memorias y buses separados para programas y datos. Tiene un archivo de registro de acceso rápido de 32 x 8 registros de trabajo de propósito general con un solo tiempo de acceso de ciclo de reloj. Los 32 registros de trabajo son una de las claves para una codificación C eficiente. Estos registros tienen la misma función que el acumulador tradicional, excepto que hay 32 de ellos. Las instrucciones aritméticas y lógicas del AVR funcionan en estos registros, por lo que ocupan menos espacio de instrucción. En un ciclo de

reloj, AVR puede enviar dos registros arbitrarios desde el archivo de registro a la ALU, realizar una operación y volver a escribir el resultado en el archivo de registro.



Las instrucciones en la memoria del programa se ejecutan con una canalización de un solo nivel. Mientras se ejecuta una instrucción, la siguiente instrucción se obtiene previamente de la memoria del programa. Este concepto permite ejecutar instrucciones en cada ciclo de reloj. La mayoría de las instrucciones AVR tienen un solo formato de palabra de 16 bits. Cada dirección de memoria de programa contiene una instrucción de 16 o 32 bits.

AVR CCG

AVR GCC proporciona varios niveles de optimización. Son -O0, -O1, -O2, -O3 y -Os. En cada nivel, hay diferentes opciones de optimización habilitadas, excepto -O0 que significa que no hay optimización. Además de las opciones habilitadas en los niveles de optimización, también puede habilitar opciones de optimización separadas para obtener una optimización específica. El manual

de la colección de compiladores GNU tiene una lista completa de opciones y niveles de optimización.

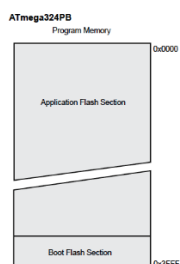
Aparte del AVR GCC, se necesitan muchas otras herramientas trabajando juntas para producir la aplicación ejecutable final para el microcontrolador AVR. El grupo de herramientas se denomina cadena de herramientas. Dentro de la cadena de herramientas AVR, AVR Libc Library, que proporciona muchas de las mismas funciones que se encuentran en una biblioteca C estándar y muchas funciones de biblioteca adicionales específicas para un AVR. Además, la biblioteca proporciona el código de inicio básico que necesitan la mayoría de las aplicaciones. Consulte el Manual de referencia de AVR Libc para obtener más detalles.

Memoria de lectura mientras se escribe AVR (gestor de arranque)

En muchos dispositivos AVR ® , la sección del cargador de arranque de la memoria del programa es compatible con la autoprogramación real de lectura mientras escribe. Esta función permite actualizaciones de software de aplicaciones flexibles controladas por la MCU mediante un programa Boot Loader residente en Flash. El programa Boot Loader puede usar cualquier interfaz de datos disponible y protocolo asociado para leer código y escribir (programar) ese código en la memoria Flash o leer el código de la memoria del programa. El código del programa dentro de la sección del cargador de arranque tiene la capacidad de escribir en todo el Flash, incluida la memoria del cargador de arranque. El Boot Loader puede incluso modificarse a sí mismo, y también puede borrarse del código si la función ya no es necesaria.

Memoria flash

La memoria Flash está organizada en dos secciones principales: la sección de la aplicación y la sección del cargador de arranque. El tamaño de las diferentes secciones lo configuran los Fusibles BOOTSZ. Estas dos secciones pueden tener diferentes niveles de protección ya que tienen diferentes conjuntos de bits de bloqueo.



Sección de aplicación

La sección Aplicación es la sección de Flash que se utiliza para almacenar el código de la aplicación. El nivel de protección para la sección Aplicación se puede seleccionar mediante los bits de bloqueo de arranque de la aplicación (bits de bloqueo de arranque 0). La sección de la aplicación nunca puede almacenar ningún código del cargador de arranque, ya que la instrucción Store Program Memory (SPM) está deshabilitada cuando se ejecuta desde la sección de la aplicación.

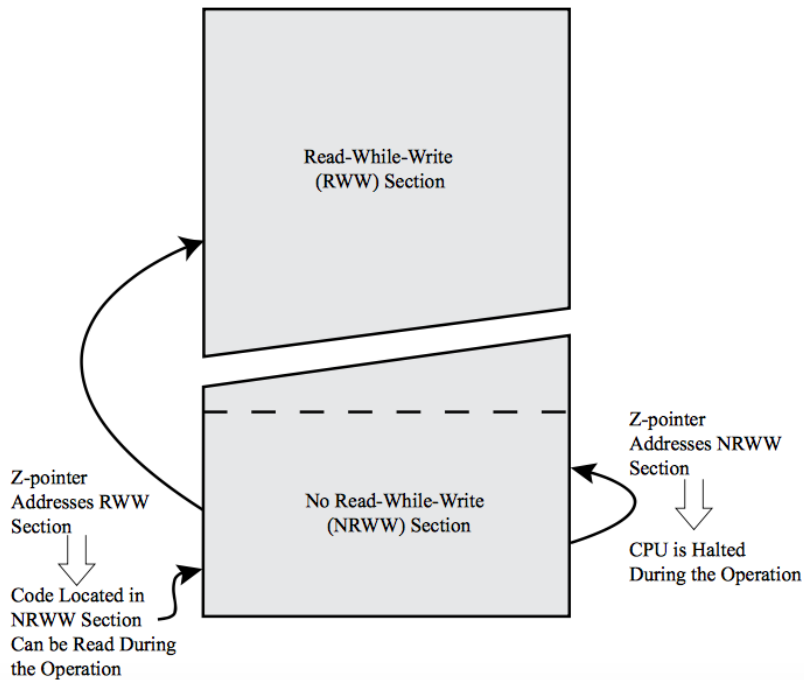
Sección del cargador de arranque (BLS)

Si bien la sección Aplicación se usa para almacenar el código de la aplicación, el software Boot Loader debe estar ubicado en el BLS ya que la instrucción SPM puede iniciar una programación cuando se ejecuta desde el BLS únicamente. La instrucción SPM puede acceder a todo el Flash, incluido el propio BLS. El nivel de protección para la sección del cargador de arranque se puede seleccionar mediante los bits de bloqueo del cargador de arranque. El ejemplo que se muestra es de la hoja de datos AVRmega328PB.

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application Section	Boot Reset Address (Start Boot Loader Section)
1	1	256 words	4	0x0000 - 0x3EFF	0x3F00 - 0x3FFF	0x3EFF	0x3F00
1	0	512 words	8	0x0000 - 0x3DFF	0x3E00 - 0x3FFF	0x3DFF	0x3E00
0	1	1024 words	16	0x0000 - 0x3BFF	0x3C00 - 0x3FFF	0x3BFF	0x3C00
0	0	2048 words	32	0x0000 - 0x37FF	0x3800 - 0x3FFF	0x37FF	0x3800

Secciones flash de lectura mientras escribe (RWW) y sin lectura mientras escribe (NRWW)

Además de las dos secciones configurables por BOOTSZ Fuses como se describió anteriormente, el flash también se divide en dos secciones fijas, la sección de lectura durante la escritura (RWW) y la sección sin lectura durante la escritura (NRWW).



La principal diferencia entre las dos secciones es:

Al borrar o escribir una página ubicada dentro de la sección RWW, la sección NRWW se puede leer durante la operación.

Al borrar o escribir una página ubicada dentro de la sección NRWW, la CPU se detiene durante toda la operación.

El límite entre las secciones RWW y NRWW se define en la ficha técnica del dispositivo. Estos son los ajustes para el AVRmega328PB.

Section	Pages	Address
Read-While-Write section (RWW)	224	0x0000 - 0x37FF
No Read-While-Write section (NRWW)	32	0x3800 - 0x3FFF

Si el dispositivo AVR es compatible con RWW o si el dispositivo AVR se detiene durante una actualización del software Boot Loader depende de la dirección que se esté programando.

El software del usuario nunca puede leer ningún código ubicado dentro de la sección RWW durante una operación del software Boot Loader. La sintaxis "Sección de lectura mientras se escribe" se refiere a qué sección se está programando (borrando o escribiendo), no a qué sección se está leyendo durante una actualización de software del cargador de arranque.

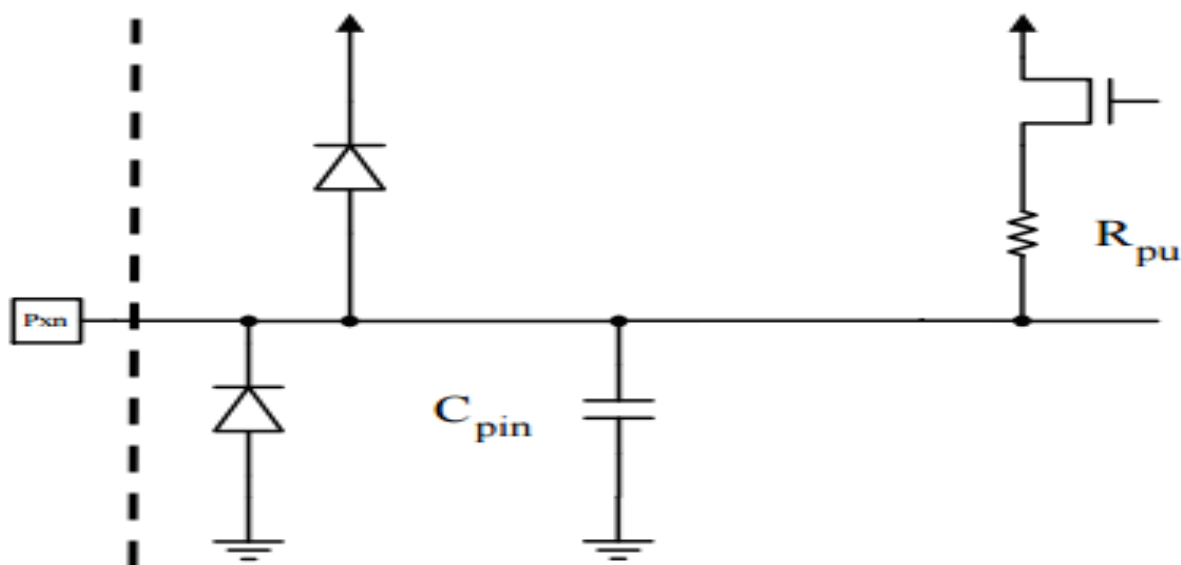
Puertos de entrada/salida digital en AVR

Los microcontroladores AVR® de 8 bits controlan las aplicaciones a través de sus pines de entrada y salida (E/S) digitales. Estos pines pueden monitorear cualquier voltaje presente como una entrada de alta impedancia y suministrar o absorber corriente como una salida digital de alto o bajo voltaje. Estos pines generalmente se organizan en grupos de ocho y se denominan puerto. El AVR usa el alfabeto para nombrar estos puertos, por ejemplo: PortA, PortB, etc. Los pines de PortA se denominan PA0 - PA7.

Visión general

Todos los puertos AVR tienen verdadera funcionalidad de lectura, modificación y escritura cuando se usan como puertos de E/S digitales generales. Esto significa que la dirección de un pin de puerto se puede cambiar sin cambiar involuntariamente la dirección de cualquier otro. Lo mismo se aplica al cambiar el valor del variador (si está configurado como salida) o al habilitar/deshabilitar las resistencias pull-up (si está configurado como entrada). Cada búfer de salida tiene características de unidad simétricas con alta capacidad de fuente y sumidero.

El controlador de pines es lo suficientemente robusto como para controlar pantallas LED directamente. Todos los pines del puerto tienen resistencias pull-up seleccionables individualmente con una resistencia invariable de voltaje de suministro. Todos los pines de E/S tienen diodos de protección tanto para VCC como para tierra, como se indica en la figura.



Resumen breve del puerto de E/S del AVR

FUENTE YOUTUBE: <https://youtu.be/bDPdrWS-YUc>

Configuración de los pines digitales de E/S

Cada puerto consta de tres registros:

- **DDRx** : registro de dirección de datos
- **PORTx** – Registro de salida de clavijas
- **PINx** : registro de entrada de PIN

donde x = Nombre del puerto (A, B, C o D)

Name: DDRB

Offset: 0x24

Reset: 0x00

Property: When addressing as I/O Register: address offset is 0x04

Bit	7	6	5	4	3	2	1	0
	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – DDRBn: Port B Data Direction [n = 7:0]

□

Name: PORTB

Offset: 0x25

Reset: 0x00

Property: When addressing as I/O Register: address offset is 0x05

Bit	7	6	5	4	3	2	1	0
	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – PORTBn: Port B Data [n = 0:7]

Name: PINB

Offset: 0x23

Reset: N/A

Property: When addressing as I/O Register: address offset is 0x03

Bit	7	6	5	4	3	2	1	0
	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	x	x	x	x	x	x	x	x

Bits 7:0 – PINBn: Port B Input Pins Address [n = 7:0]

Estos registros determinan la configuración de las entradas y salidas digitales. Los pines de E/S también se pueden compartir con periféricos internos. Por ejemplo, el convertidor analógico a digital (ADC) se puede conectar al pin de E/S en lugar de ser un pin digital. En este caso, los registros de pines de E/S lo configuran como una entrada de alta impedancia de tres estados.

Bits de registro

- Se accede a los bits DDxn en la dirección de E/S DDRx
- Bits PORTxn en la dirección de E/S PORTx
- Bits PINxn en la dirección de E/S PINx

Donde n = número de bit de pin en el registro de puerto

DDxn

Los bits DDxn en el Registro DDRx seleccionan la dirección de este pin. Si DDxn se escribe en '1', Pxn se configura como pin de salida. Si DDxn se escribe en '0', Pxn se configura como pin de entrada.

PUERTOxn

Los bits PORTxn en el registro PORTx tienen dos funciones. Pueden controlar el estado de salida de un pin y la configuración de un pin de entrada.

Como

salida:

si se escribe un '1' en el bit cuando el pin está configurado como un pin de salida, el pin del puerto se

eleva. Si se escribe un '0' en el bit cuando el pin está configurado como un pin de salida, el pin del puerto está bajo.

Como

entrada:

si se escribe un '1' en el bit cuando el pin está configurado como pin de entrada, se activa la resistencia pull-up. Si se escribe un '0' en el bit cuando el pin está configurado como pin de entrada, el pin del puerto tiene tres estados.

PINxn

Los bits PINxn en el registro PINx se utilizan para leer datos del pin del puerto. Cuando el pin está configurado como una entrada digital (en el registro DDRx) y el pull-up está habilitado (en el registro PORTx), el bit indicará el estado de la señal en el pin (alto o bajo).

Nota: si un puerto se convierte en una salida, la lectura del registro PINx le proporcionará datos que se han escrito en los pines del puerto.

Como entrada de tres estados:
cuando el registro PORTx deshabilita la resistencia pull-up, la entrada será de tres estados, dejando el pin flotando. Cuando se deja en este estado, incluso una pequeña carga estática presente en los objetos circundantes puede cambiar el estado lógico del pin. Si intenta leer el bit correspondiente en el registro pin, no se puede predecir su estado.

Ejemplos

Todos los pines PORTA configurados como entradas con pull-ups habilitados y luego leen datos de PORTA:

```
DDRA = 0x00;    //make PORTA all inputs
PORTA = 0xFF;   //enable all pull-ups
data = PINA;    //read PORTA pins into variable data
```

PORTB configurado para entradas de tres estados:

```
DDRB = 0x00;    //make PORTB all inputs
PORTB = 0x00;   //disable pull-ups and make all pins tri-state
```

PORTA nybble inferior configurado como salidas, nybble superior como entradas con pull-ups habilitados:

```
DDRA = 0x0F;    //lower pins output, higher pins input
PORTA = 0xF0;   //output pins set to 0, input pins enable pull-ups
```

Está disponible un proyecto de ejemplo que controla un pin de E/S junto con una depuración simple.

Proyecto de E/S digital en AVR Xplained 328PB

Proyecto de ejemplo mencionado en el video

[| Ejemplo de E/S con referencia de vídeo]]

Consejos y trucos

Cambio entre entrada y salida

Al cambiar entre tres estados ($\{DDxn, PORTxn\} = 0b00$) y salida alta ($\{DDxn, PORTxn\} = 0b11$), un estado intermedio con pull-up habilitado ($\{DDxn, PORTxn\} = 0b01$) o salida baja ($\{DDxn, PORTxn\} = 0b10$) debe ocurrir.

Normalmente, el estado habilitado de pull-up es completamente aceptable, ya que un entorno de alta impedancia no notará la diferencia entre un controlador alto fuerte y un pull-up. Si este no es el caso, el bit PUD en el registro MCUCR se puede configurar para deshabilitar todos los pull-ups en todos los puertos.

Cambiar entre entrada con pull-up y salida baja genera el mismo problema. Debe utilizar el estado triple ($\{DDxn, PORTxn\} = 0b00$) o el estado alto de salida ($\{DDxn, PORTxn\} = 0b11$) como paso intermedio.

DDxn	PORTxn	PUD (in MCUCR)	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)

Deshabilitar anulación de pull-ups

El bit **PUD** Pull-up Disable en el registro **MCUCR** puede anular la configuración pull-up de **DDRx** y **PORTx** .

Name: MCUCR

Offset: 0x55

Reset: 0x00

Property: When addressing as I/O Register: address offset is 0x35

Bit	7	6	5	4	3	2	1	0
		BODS	BODSE	PUD			IVSEL	IVCE
Access		R/W	R/W	R/W			R/W	R/W
Reset		0	0	0			0	0



Cuando este bit se escribe en uno, los pull-ups en los puertos de E/S se deshabilitan incluso si los **registros DDxn** y **PORTxn** están configurados para habilitar los pull-ups (**{DDxn, PORTxn} = 0b01**) .

Alternar un pin de E/S

Escribir un '1' en **PINxn** cambia el valor de **PORTxn** independientemente del valor de **DDRxn**. La instrucción de ensamblaje **SBI** se puede usar para alternar un solo bit en un puerto.

Pines desconectados

Si algunos pines no se utilizan, le recomendamos que se asegure de que estos pines tengan un nivel definido, aunque la mayoría de las entradas digitales estén deshabilitadas en los modos de suspensión profunda. Deben evitarse las entradas flotantes para reducir el consumo de corriente en todos los demás modos en los que las entradas digitales están habilitadas (reinicio, modo activo y modo inactivo).

El método más simple para garantizar un nivel definido de un pin no utilizado es habilitar el pull-up interno. En este caso, el pull-up se desactivará durante el reinicio. Si el bajo consumo de energía durante el reinicio es importante, le recomendamos que utilice un pull-up o pull-down externo. NO se recomienda conectar los pines no utilizados directamente a **VCC** o **GND**, ya que esto puede causar corrientes excesivas si el pin se configura accidentalmente como una salida.

Modos de suspensión de bajo consumo de AVR®

Los microcontroladores AVR® de 8 bits incluyen varios modos de suspensión que permiten que la aplicación apague los módulos no utilizados en la MCU, ahorrando así energía. El dispositivo AVR proporciona varios modos de suspensión que le permiten adaptar el consumo de energía a los requisitos de la aplicación.

Hay cinco modos de suspensión para seleccionar:

- Modo inactivo
- Corriente cortada
- Ahorro de energía
- Apoyar
- Espera extendida

Para ingresar a cualquiera de los modos de suspensión, el bit de habilitación de suspensión en el **registro de control del modo de suspensión (SMCR.SE)** debe escribirse en '1' y debe ejecutarse una instrucción SLEEP. Los **bits de selección de modo de reposo (SMCR.SM[2:0])** seleccionan qué modo de reposo (inactivo, apagado, ahorro de energía, espera o espera extendida) se activará mediante la instrucción SLEEP. Las opciones de suspensión que se muestran son de ATmega328PB.

Name: SMCR
Offset: 0x53
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x33

Bit	7	6	5	4	3	2	1	0
					SM2	SM1	SM0	SE
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – SM2: Sleep Mode Select 2

The SM[2:0] bits select between the five available sleep modes.

Table 14-2 Sleep Mode Select

SM2,SM1,SM0	Sleep Mode
000	Idle
001	
010	Power-down
011	Power-save
100	Reserved
101	Reserved
110	Standby ⁽¹⁾
111	Extended Standby ⁽¹⁾



Interrupciones

Si ocurre una interrupción habilitada mientras la MCU está en modo de suspensión, la MCU se activa. Luego, la MCU se detiene durante cuatro ciclos además del tiempo de inicio, ejecuta la rutina de interrupción y reanuda la ejecución desde la instrucción que sigue a SLEEP. El contenido del archivo de registro y SRAM no se altera cuando el dispositivo se despierta del modo de suspensión.

Reiniciar

Si se produce un reinicio durante el modo de suspensión, la MCU se activa y se ejecuta desde el vector de reinicio.

Breve resumen del modo de suspensión del AVR

FUENTE YOUTUBE: <https://youtu.be/Fyspr40KcMU>

Descarga del proyecto: [proyecto de muestra mencionado en el video](#)

Modo inactivo

Cuando los bits **SM[2:0]** se escriben en '000', la instrucción **SLEEP** hace que la MCU entre en modo inactivo, lo que detiene la CPU pero permite que la interfaz periférica en serie (SPI), el receptor/transmisor síncrono/asíncrono universal (USART), Comparador Analógico, Interfaz Serial de 2 hilos, Temporizador/Contadores, Watchdog y el sistema de interrupción para continuar operando. Este modo de suspensión básicamente detiene el **reloj de la CPU (clkCPU)** y el **reloj Flash (clkFLASH)**, mientras permite que se ejecuten los otros relojes. El modo inactivo permite que la MCU se despierte de las interrupciones externas, así como de las internas, como el desbordamiento del temporizador y las interrupciones de transmisión completa de USART. Si no se requiere la activación de la interrupción del comparador analógico, el comparador analógico se puede apagar configurando el bit ACD en el registro de control y estado del comparador analógico - ACSR. Esto reducirá el consumo de energía en el modo inactivo.

Modo de apagado

Cuando los bits **SM[2:0]** se escriben en '010', la instrucción **SLEEP** hace que la MCU ingrese al modo de apagado. En este modo, el Oscilador externo se detiene, mientras que las interrupciones externas, el reloj

de dirección de la Interfaz Serial de 2 hilos y el Watchdog continúan operando (si está habilitado). Solo uno de estos eventos puede despertar la MCU:

- Restablecimiento externo
- Restablecimiento del sistema de vigilancia
- Interrupción de vigilancia
- Restablecimiento de oscurecimiento
- Coincidencia de dirección de interfaz serial de 2 hilos
- Interrupción de nivel externo en INT
- Interrupción por cambio de pin

Este modo de suspensión básicamente detiene todos los relojes generados, lo que permite la operación de módulos asíncronos únicamente.

Nota: Si se usa una interrupción activada por nivel para activarse después de un apagado, el nivel requerido debe mantenerse el tiempo suficiente para que la MCU complete la activación para activar la interrupción de nivel. Si el nivel desaparece antes de que finalice el tiempo de inicio, la MCU aún se activará, pero no se generará ninguna interrupción. El tiempo de inicio está definido por el **temporizador de inicio (SUT)** y los fusibles de selección de **reloj (CKSEL)** .

Cuando se activa desde el modo de apagado, hay un retraso desde que ocurre la condición de activación hasta que la activación se hace efectiva. Esto permite que el reloj se reinicie y se establezca después de haber sido detenido. El período de activación está definido por los mismos fusibles **CKSEL** que definen el **período de tiempo de espera de restablecimiento** .

Modo ahorro de energía

Cuando los **bits SM[2:0]** se escriben en **011** , la instrucción SLEEP hace que la MCU ingrese al modo de ahorro de energía. Este modo es idéntico al Apagado, con una excepción: si el Temporizador/Contador2 está habilitado, seguirá funcionando durante la suspensión.

El dispositivo puede despertarse del evento de desbordamiento del temporizador o de comparación de salida del temporizador/contador2 si los bits correspondientes de habilitación de interrupción del temporizador/contador2 están configurados en el **registro de máscara de interrupción del temporizador (TIMSK2)** y el bit de **habilitación de interrupción global en el registro de estado (SREG)** es establecer.

Si Timer/Counter2 no está funcionando, se recomienda el modo de apagado en lugar del modo de ahorro de energía.

El Timer/Counter2 se puede cronometrar de forma sincrónica y asincrónica en el modo de ahorro de energía. Si Timer/Counter2 no está usando el reloj asíncrono, el Timer/Counter Oscillator se detiene durante la suspensión. Si Timer/Counter2 no está usando el reloj síncrono, la fuente del reloj se detiene

durante la suspensión. Incluso si el reloj síncrono está funcionando en ahorro de energía, este reloj solo está disponible para Timer/Counter2.

Modo de espera

Cuando los bits **SM[2:0]** se escriben en '110' y se selecciona una opción de reloj externo, la instrucción SLEEP hace que la MCU entre en modo de espera. Este modo es idéntico a Power-Down con la excepción de que el Oscilador sigue funcionando. Desde el modo de espera, el dispositivo se activa en seis ciclos de reloj.

Modo de espera extendido

Cuando los bits **SM[2:0]** se escriben en '111' y se selecciona una opción de reloj externo, la instrucción SLEEP hace que la MCU ingrese al modo de espera extendida. Este modo es idéntico al modo de ahorro de energía con la excepción de que el oscilador se mantiene en funcionamiento. Desde el modo de espera extendida, el dispositivo se activa en seis ciclos de reloj.

Consejos y trucos para minimizar el consumo de energía

Hay varias posibilidades a considerar cuando se trata de minimizar el consumo de energía en un sistema controlado por AVR. En general, los modos de suspensión se deben usar tanto como sea posible, y el modo de suspensión se debe seleccionar de modo que estén operando la menor cantidad posible de funciones del dispositivo. Todas las funciones que no sean necesarias deben desactivarse.

En particular, los siguientes módulos pueden necesitar una consideración especial cuando se trata de lograr el menor consumo de energía posible.

Conversor analógico a digital

Si está habilitado, el ADC estará habilitado en todos los modos de suspensión. Para ahorrar energía, el ADC debe desactivarse antes de ingresar a cualquier modo de suspensión. Cuando el ADC se apaga y se vuelve a encender, la próxima conversión será una conversión extendida.

Comparador analógico

Al ingresar al modo inactivo, el comparador analógico debe desactivarse si no se usa. Al ingresar al modo de reducción de ruido ADC, el comparador analógico debe estar desactivado. En otros modos de reposo, el comparador analógico se desactiva automáticamente. Sin embargo, si el comparador analógico está configurado para usar la referencia de voltaje interno como entrada, el comparador analógico debe

desactivarse en todos los modos de reposo. De lo contrario, se habilitará la referencia de voltaje interno, independientemente del modo de reposo.

Detector de oscurecimiento

Si la aplicación no necesita el detector Brown-Out (BOD), este módulo debe apagarse. Si el BOD está habilitado por los fusibles BODLEVEL, estará habilitado en todos los modos de suspensión y, por lo tanto, siempre consumirá energía. En los modos de suspensión más profunda, esto contribuirá significativamente al consumo total de corriente.

Referencia de voltaje interno

La referencia de voltaje interno se habilitará cuando sea necesario para la detección de Brown-Out, el comparador analógico o el convertidor de analógico a digital. Si estos módulos se deshabilitan como se describe en las secciones anteriores, la referencia de voltaje interno se deshabilitará y no consumirá energía. Cuando se vuelve a encender, el usuario debe permitir que la referencia se inicie antes de que se use la salida. Si la referencia se mantiene en modo de reposo, la salida se puede utilizar inmediatamente.

Temporizador de vigilancia

Si no se necesita el temporizador de vigilancia en la aplicación, el módulo debe apagarse. Si el temporizador de vigilancia está habilitado, estará habilitado en todos los modos de suspensión y, por lo tanto, siempre consumirá energía. En los modos de suspensión más profunda, esto contribuirá significativamente al consumo total de corriente.

Pines de puerto

Al ingresar al modo de suspensión, todos los pines del puerto deben configurarse para usar la energía mínima. Entonces, lo más importante es asegurarse de que ningún pasador impulse cargas resistivas. En los modos de suspensión en los que se detienen tanto el reloj de E/S (clkI/O) como el reloj del ADC (clkADC), los búferes de entrada del dispositivo se desactivarán. Esto asegura que la lógica de entrada no consume energía cuando no se necesita. En algunos casos, la lógica de entrada es necesaria para detectar las condiciones de activación y luego se habilitará. Consulte la sección Habilitación de entrada digital y modos de suspensión para obtener detalles sobre qué pines están habilitados. Si el búfer de entrada está habilitado y la señal de entrada se deja flotando o tiene un nivel de señal analógica cercano a $VCC/2$, el búfer de entrada usará una potencia excesiva.

Para los pines de entrada analógica, el búfer de entrada digital debe estar deshabilitado en todo momento. Un nivel de señal analógica cercano a $VCC/2$ en un pin de entrada puede causar una corriente

significativa incluso en modo activo. Los búferes de entrada digital se pueden desactivar escribiendo en los registros de desactivación de entrada digital (DIDR0 para ADC, DIDR1 para AC).

Sistema de depuración en chip

Si el sistema de depuración en chip está habilitado por el fusible DWEN y el chip entra en modo de suspensión, la fuente de reloj principal está habilitada y, por lo tanto, siempre consume energía. En los modos de suspensión más profunda, esto contribuirá significativamente al consumo total de corriente.

Registro de reducción de energía periférica del dispositivo AVR®

Los microcontroladores AVR® de 8 bits incluyen varios modos de suspensión para ahorrar energía. El dispositivo AVR también puede reducir el consumo de energía apagando el reloj, para periféricos seleccionados, a través de una configuración de registro. Ese registro se denomina **Registro de Reducción de Potencia (PRR)**.

Ejemplo: Registro desde ATmega328PB .

Escribir un uno a un bit lógico apaga el reloj periférico.

Bit	7	6	5	4	3	2	1	0
	PRTWI0	PRTIM2	PRTIM0	PRUSART1	PRTIM1	PRSPI0	PRUSART0	PRADC
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0



- PRTWI0: Reducción de potencia TWI0
- PRTIM2: Temporizador/Contador2 de reducción de potencia
- PRTIM0: Temporizador/Contador 0 de reducción de potencia
- PRUSART1: Reducción de potencia USART1
- PRTIM1: Temporizador/Contador1 de reducción de potencia
- PRSPI0: interfaz periférica serial de reducción de energía 0
- PRUSART0: Reducción de potencia USART0
- PRDC: ADC de reducción de potencia

El **PRR** proporciona un método de tiempo de ejecución para detener el reloj y seleccionar periféricos individuales. El estado actual del periférico está congelado y los registros de E/S no se pueden leer ni escribir. Los recursos utilizados por el periférico al detener el reloj seguirán comprometidos, por lo tanto, en la mayoría de los casos, el periférico debe desactivarse antes de detener el reloj. Activar un módulo, que se realiza borrando el bit en PRR, pone el módulo en el mismo estado que antes del apagado.

El **apagado del reloj PRR** se puede usar en modo inactivo y modo activo para reducir significativamente el consumo de energía general. En todos los demás modos de suspensión, el reloj ya está detenido.

Resumen breve del registro de reducción de potencia del AVR

FUENTE YOUTUBE: <https://youtu.be/S82BSPbYoVA>

Descarga del proyecto: [proyecto de muestra mencionado en el video](#)

Minimizar el consumo de energía

Hay varias posibilidades a considerar cuando se trata de minimizar el consumo de energía en un sistema controlado por AVR. En general, los modos de suspensión se deben usar tanto como sea posible, y el modo de suspensión se debe seleccionar de modo que estén operando la menor cantidad posible de funciones del dispositivo. Todas las funciones que no sean necesarias deben desactivarse. En particular, los siguientes módulos pueden necesitar una consideración especial cuando se trata de lograr el menor consumo de energía posible:

Conversor analógico a digital

Si está habilitado, el ADC estará habilitado en todos los modos de suspensión. Para ahorrar energía, el ADC debe desactivarse antes de ingresar a cualquier modo de suspensión. Cuando el ADC se apaga y se vuelve a encender, la próxima conversión será una conversión extendida.

Comparador analógico

Al ingresar al modo inactivo, el comparador analógico debe desactivarse si no se usa. Al ingresar al modo de reducción de ruido ADC, el comparador analógico debe estar desactivado. En otros modos de reposo, el comparador analógico se desactiva automáticamente. Sin embargo, si el comparador analógico está configurado para usar la referencia de voltaje interno como entrada, el comparador analógico debe desactivarse en todos los modos de reposo. De lo contrario, se habilitará la referencia de voltaje interno, independientemente del modo de reposo.

Detector de oscurecimiento

Si la aplicación no necesita el detector Brown-Out (BOD), este módulo debe apagarse. Si el BOD está habilitado por los fusibles BODLEVEL, estará habilitado en todos los modos de suspensión y, en consecuencia, siempre consumirá energía. En los modos de suspensión más profunda, esto contribuirá significativamente al consumo total de corriente.

Referencia de voltaje interno

La referencia de voltaje interno se habilitará cuando lo necesite el detector Brown-Out, el comparador analógico o el convertidor de analógico a digital. Si estos módulos se deshabilitan como se describe en las secciones anteriores, la referencia de voltaje interno se deshabilitará y no consumirá energía. Cuando se vuelve a encender, el usuario debe permitir que la referencia se inicie antes de que se use la salida. Si la referencia se mantiene en modo de reposo, la salida se puede utilizar inmediatamente.

Temporizador de vigilancia

Si no se necesita el temporizador de vigilancia en la aplicación, el módulo debe apagarse. Si el temporizador de vigilancia está habilitado, estará habilitado en todos los modos de suspensión y, por lo tanto, siempre consumirá energía. En los modos de suspensión más profunda, esto contribuirá significativamente al consumo total de corriente.

Pines de puerto

Al ingresar al modo de suspensión, todos los pines del puerto deben configurarse para usar la energía mínima. La consideración más importante es asegurarse de que ningún pasador impulse cargas resistivas. En los modos de suspensión en los que se detienen tanto el reloj de E/S (clkI/O) como el reloj del ADC (clkADC), los búferes de entrada del dispositivo se desactivarán. Esto asegura que la lógica de entrada no consume energía cuando no se necesita. En algunos casos, la lógica de entrada es necesaria para detectar las condiciones de activación y luego se habilitará. Si el búfer de entrada está habilitado y la señal de entrada se deja flotando o tiene un nivel de señal analógica cercano a $VCC/2$, el búfer de entrada usará una potencia excesiva.

Para los pines de entrada analógica, el búfer de entrada digital debe estar deshabilitado en todo momento. Un nivel de señal analógica cercano a $VCC/2$ en un pin de entrada puede causar una corriente significativa incluso en modo activo. Los búferes de entrada digital se pueden desactivar escribiendo en los registros de desactivación de entrada digital (DIDR0 para ADC, DIDR1 para AC).

Sistema de depuración en chip

Si el sistema de depuración en chip está habilitado por el fusible DWEN y el chip entra en modo de suspensión, la fuente de reloj principal está habilitada y siempre consume energía. En los modos de suspensión más profunda, esto contribuirá significativamente al consumo total de corriente.

Fusibles AVR®

Los fusibles AVR ® son las ubicaciones en la memoria no volátil que definen la configuración de hardware de un dispositivo AVR.

Los fusibles se colocan en una sección seleccionada de la memoria y constan de unos pocos registros. Cada bit del registro representa una configuración de fusible diferente. Puede encontrar información detallada sobre qué fusibles están disponibles en los diferentes modos de programación y sus funciones en la hoja de datos del dispositivo. La velocidad del reloj de instrucciones, el temporizador de vigilancia y el modo de depuración son solo algunas de las configuraciones de fusibles disponibles en la mayoría de los dispositivos. Los fusibles se cambian en el momento de la programación con un programador conectado, como el depurador/programador ICE , utilizando Atmel Studio 7 IDE. Se traban en su lugar después de la programación y también al encender el dispositivo.

A menos que se preste una cuidadosa consideración a qué bits Fuse están programados, es fácil para el programador novato bloquear su dispositivo (es decir, hacer que la MCU no sea programable/no depurable en su circuito de aplicación).

Resumen breve de fusibles AVR

FUENTE YOUTUBE: <https://youtu.be/YG5QypvIMLQ>

Programación de fusibles

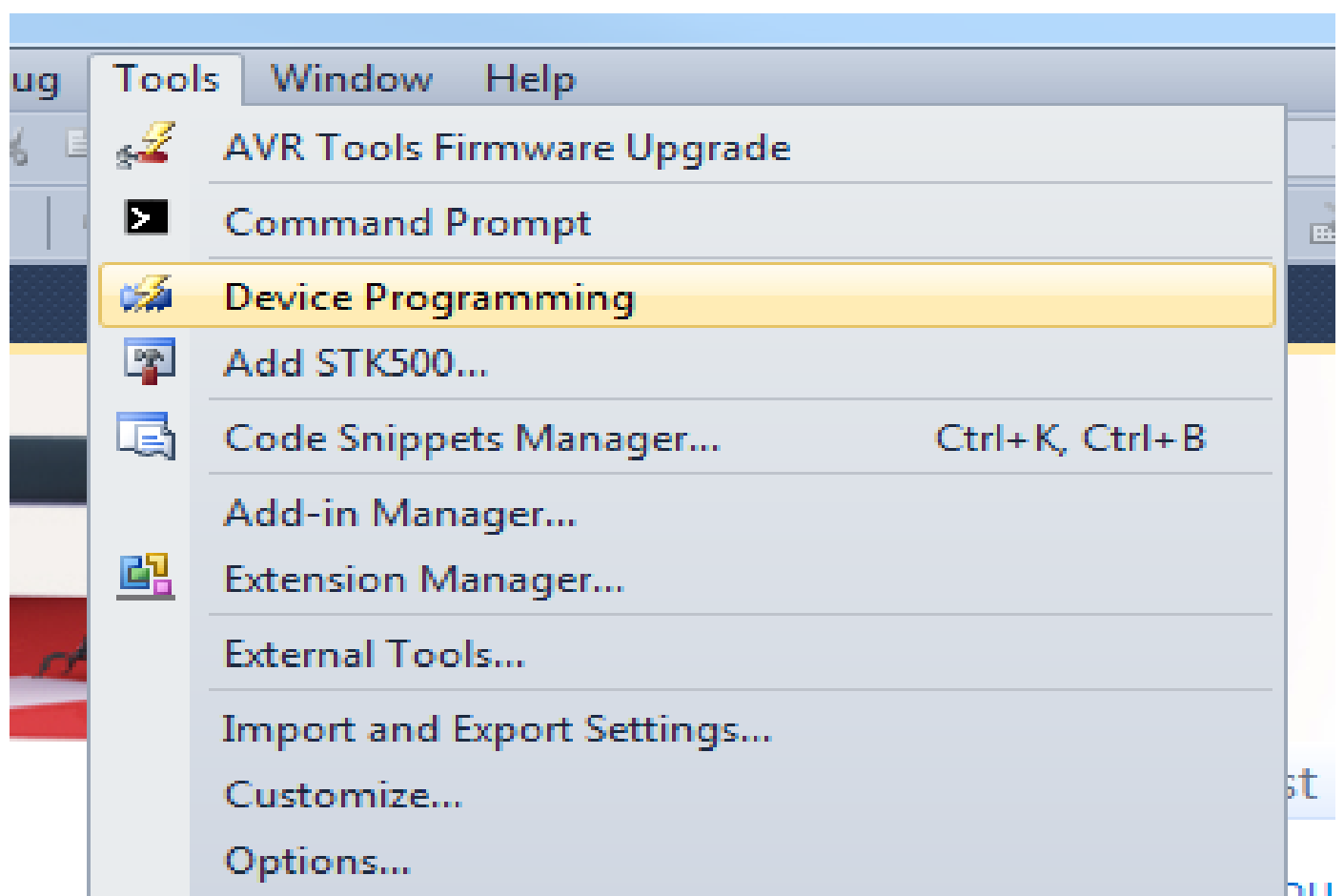
La ventana **Programación de dispositivos** (también conocida como cuadro de diálogo de programación) le brinda el control de nivel más bajo sobre las herramientas de depuración y programación. Con él, puede programar las diferentes memorias, fusibles y bits de bloqueo del dispositivo, borrar memorias y escribir firmas de usuarios. También puede ajustar algunas de las propiedades del kit de inicio, como los generadores de voltaje y de reloj.

Se puede acceder al cuadro de diálogo de programación desde el icono **Programación de dispositivos** en la barra de herramientas estándar o en la selección del menú desplegable **Herramientas > Programación de dispositivos** .

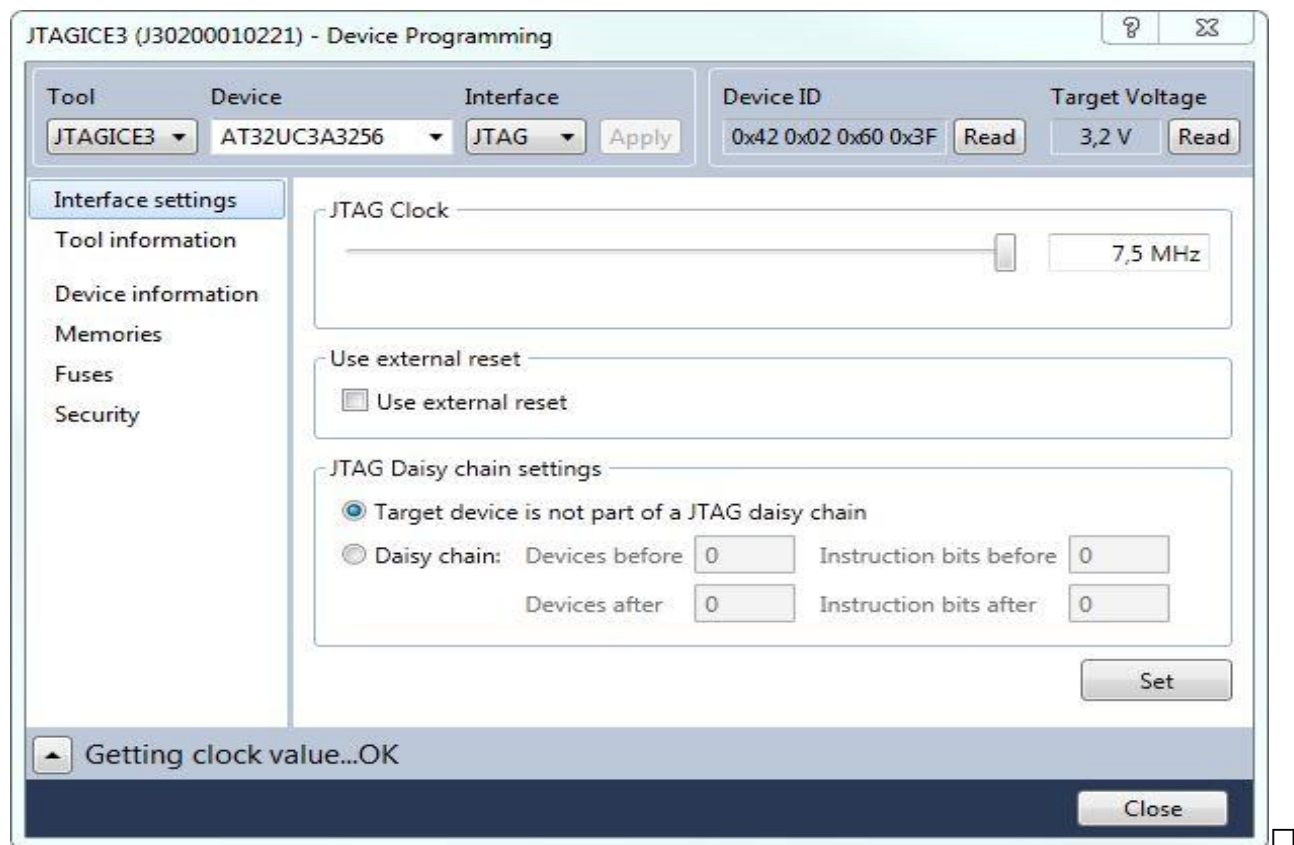
Icono de programación de dispositivos



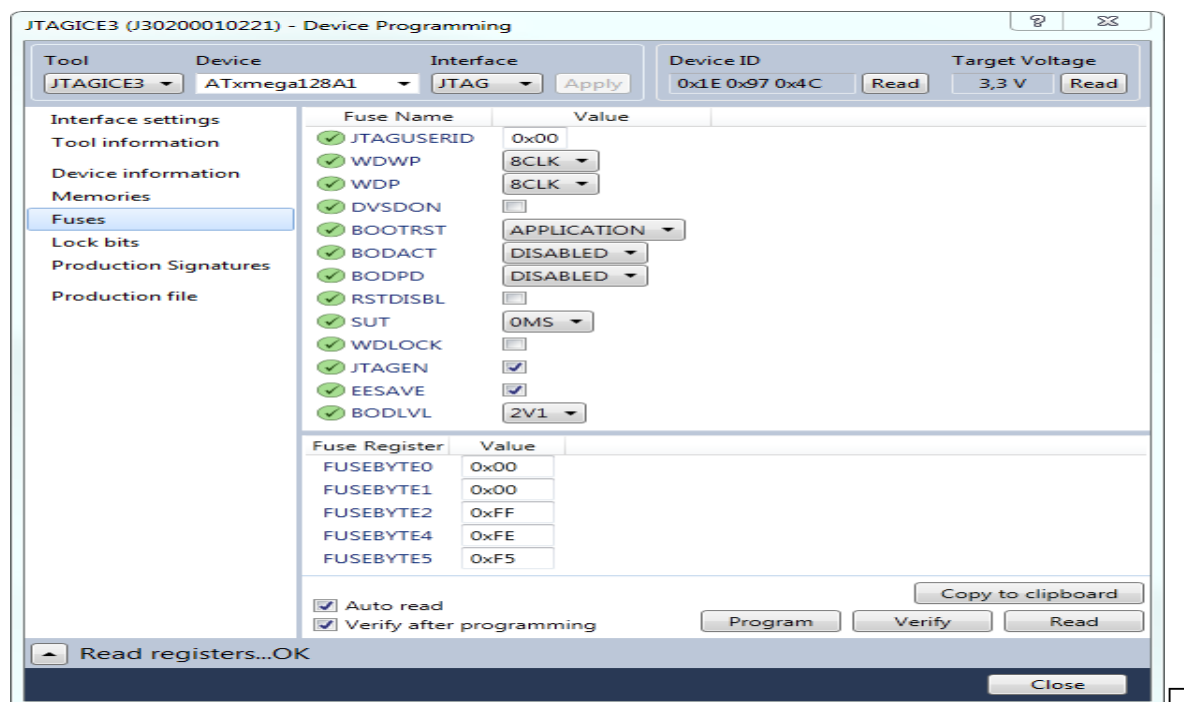
Selección de menú



Aparecerá la ventana Programación del dispositivo con varias opciones de programación enumeradas a la izquierda .



Seleccione la opción **Fusibles** y aparecerá la página de fusibles mostrando los fusibles del dispositivo seleccionado. Los ajustes de fusibles se presentan como casillas de verificación o como listas desplegables. La configuración del registro de fusibles también aparece en el panel inferior como valores hexadecimales.



Leer

Presione el botón **Leer** , en la esquina inferior derecha, para leer el valor actual de los fusibles. Si la casilla "Lectura automática" está marcada, la configuración del fusible se leerá desde el dispositivo cada vez que ingrese a la página de fusibles.

Cambio

La configuración del fusible se puede cambiar a través de los cuadros desplegables. Algunas selecciones requieren que se marque o desmarque una casilla.

Nota: Un bit de fusible que se selecciona se establece en un "0" en el registro.

Programa

Después de realizar cualquier cambio en la configuración, presione el botón **Programar** para escribir la configuración actual del fusible en el dispositivo. Si la casilla "Verificar después de la programación" está marcada, la configuración se verificará después de que se complete una operación de programación.

Pestillo

Los valores de los fusibles se bloquean cuando el dispositivo ingresa al modo de programación, y los cambios de los valores de los fusibles no tendrán efecto hasta que la pieza abandone el modo de programación. Esto no aplica para el Fusible EESAVE, el cual se hará efectivo una vez sea programado. Los fusibles también se traban en el encendido en modo normal.

Agregar configuraciones de fusibles al archivo de producción .elf

La API Fuse le permite a un usuario especificar la configuración del fusible para el dispositivo AVR específico para el que está compilando. Estos ajustes de fusibles se colocarán en una sección especial en el archivo de salida ELF después de la vinculación. Las herramientas de programación pueden aprovechar la información de fusibles incrustada en el archivo ELF, extrayendo esta información y determinando si es necesario programar los fusibles antes de programar las memorias Flash y EEPROM. Esto también permite que un solo archivo ELF contenga toda la información necesaria para programar un AVR.

Agregar configuraciones de fusibles a main.c

Los ajustes de fusibles se pueden agregar a main.c mediante el [procedimiento que se describe aquí](#) .

Conversión de formato Studio .elf a .hex para programación con MPLAB® X Tools

Si observa la salida de compilación en Studio IDE, verá cómo se crea el archivo .hex a partir del archivo .elf, por ejemplo:

```
"E:\Archivos de programa (x86)\Atmel\Studio\7.0\toolchain\avr8\avr8-gnu-toolchain\bin\avr-objcopy.exe" -O ihex -R .eeprom -R .fuse -R .lock -R .firma -R .usuario_firmas "ATtiny416.elf" "ATtiny416.hex"
```

Esto significa: haga el .hex pero deseche los archivos .eeprom, .fuse, .lock, .signature y .user_signature.

Si desea un archivo HEX con esas secciones sobrantes para usar con las herramientas de programación MPLAB X IDE, realice un paso posterior a la compilación, por ejemplo:

```
"$(ToolchainDir)\avr-objcopy.exe" -O ihex -R .eeprom -R .lock -R .signature -R user_signatures "$(OutputDirectory)\$(OutDir)\$(OutputFileName)$(OutputFileExtension)" "$(Directorio de salida)\$(Nombre del archivo de salida).with-fuse.hex"
```

El motivo de la compilación posterior es que Atmel Studio considera .elf como el archivo de producción, al igual que el 'entorno AVR clásico', mientras que el entorno MPLAB X usa archivos .hex.

Bloquear un dispositivo

Los siguientes bytes de fusible (y la configuración predeterminada de bits de fusible) se muestran para el ATmega328PB . Se resaltan los ajustes de bits de fusibles clave que podrían bloquear el dispositivo:

FUSIBLE BAJO Byte

R/P-0	R/P-1	R/P-1	R/P-0	R/P-0	R/P-0	R/P-1	R/P-0
CKDIV8	PAGAR	SUT1	SUT0	CKSEL3	CKSEL2	CKSEL1	CKSEL0
bit 7							bit 0

- **CKSEL_n [n=3:0]** : selecciona la fuente de reloj para el reloj del sistema.
- **SUT_n [n=1:0]**: selecciona el período de retraso desde que se libera el restablecimiento externo (ya no está activo) hasta que se libera el restablecimiento interno.
- **CKOUT** – Habilita la salida del reloj en PB0.

- CKDIV8: configura el reloj de la CPU para que se escale previamente en 8.

- Fusible bit = 1 no está programado (**inactivo/deshabilitado**).
- Fusible bit = 0 está programado (**activo/habilitado**).

FUSIBLE ALTO Byte

R/P-1 R/P-1 R/P-0 R/P-1 R/P-1 R/P-0 R/P-0 R/P-1

RSTDISBL	DWEN	SPIEN	WDTON	ESGUARDAR	BOTASZ1	BOTASZ0	BOOTRST
----------	------	-------	-------	-----------	---------	---------	---------

bit 7

bit 0

- BOOTRST: si usa un cargador de arranque para flashear MCU, este bit debe estar habilitado.
- BOOTSZn [n=1:0]: estos bits seleccionan el tamaño de la sección del gestor de arranque.
- EESAVE: excluye la EEPROM durante un procedimiento de borrado de chip.
- WDTON: habilite el temporizador de vigilancia por HW.
- **SPIEN** : activa/desactiva el modo de programación en serie en circuito (ISP).
- **DWEN** : habilitar/deshabilitar la interfaz de depuración DebugWire.
- **RSTDISBL** : habilitar/deshabilitar el uso del pin nRESET como IO.

FUSIBLE EXTENDIDO Byte

U-1 U-1 U-1 U-1 U-1 R/P-1 R/P-1 R/P-1

-	-	-	-	-	BODLEVEL2	BODLEVEL1	BODLEVEL0
---	---	---	---	---	-----------	-----------	-----------

bit 7

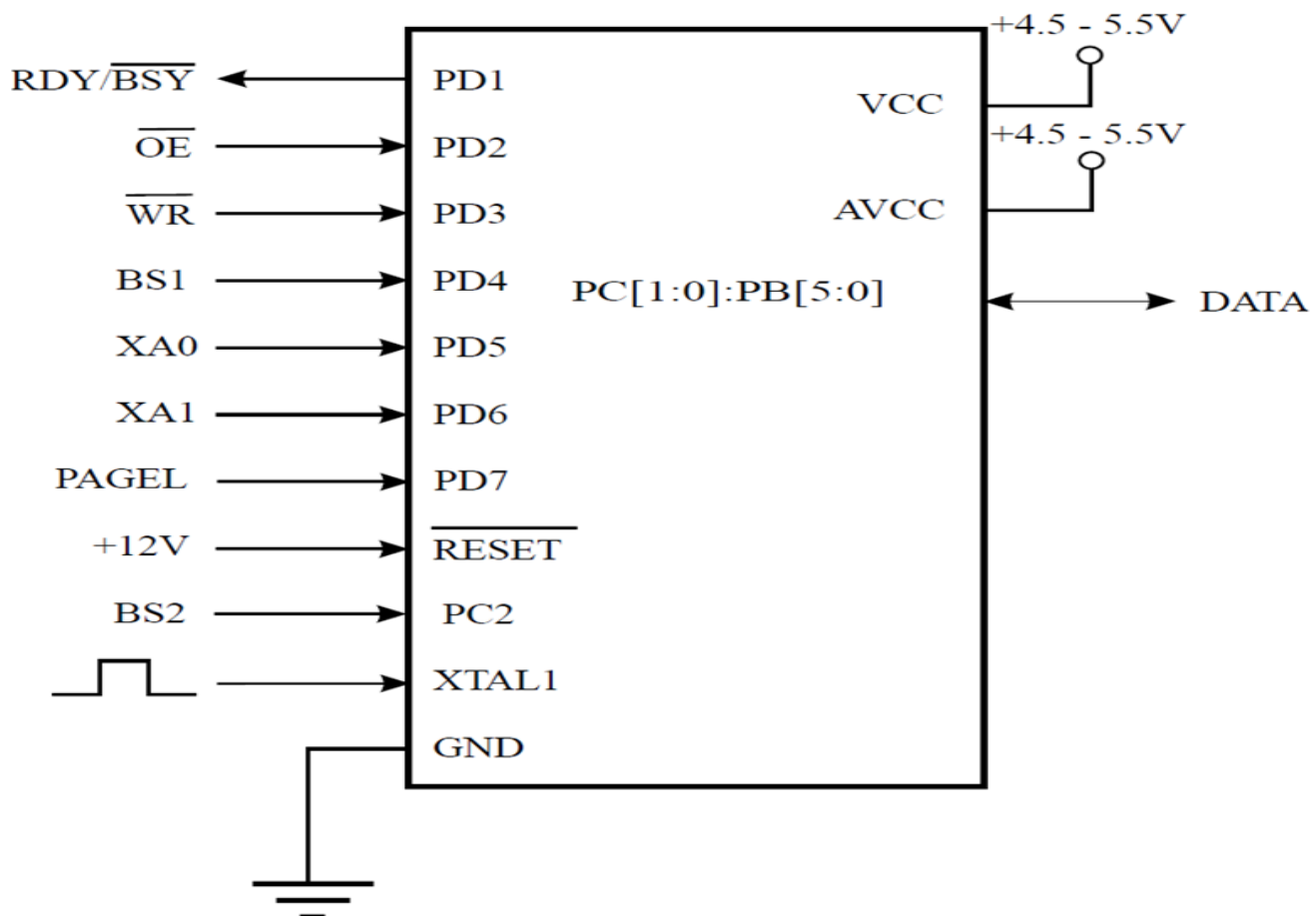
bit 0

- BODLEVELn [n=2:0] - Seleccionan el nivel de voltaje de caída de voltaje cuando el suministro de VDD ya no es adecuado para la operación y la MCU se reinicia.

Interfaces de programación AVR®

HVPP

En primer lugar, es importante comprender que todos los dispositivos Tiny y Mega basados en AVR (consulte las excepciones a continuación) incluyen una interfaz de programación **HVPP** (programación en paralelo de alto voltaje) o una interfaz de programación **HVSP** (programación en serie de alto voltaje). Ambos requieren la aplicación de un "alto voltaje" (12 V) al pin de reinicio y ambos requieren acceso a una gran cantidad de pines. La interfaz HVPP requiere acceso a al menos 16 pines, mientras que la interfaz HVSP requiere acceso a al menos 8 pines. Por esas razones, estas interfaces se utilizan principalmente para la programación de producción de los dispositivos. El siguiente esquema muestra las conexiones HVPP requeridas para el **ATmega328PB** (consulte la sección 33.7 en la hoja de datos):



ATmega328PB HVPP Connections



Las buenas noticias ...

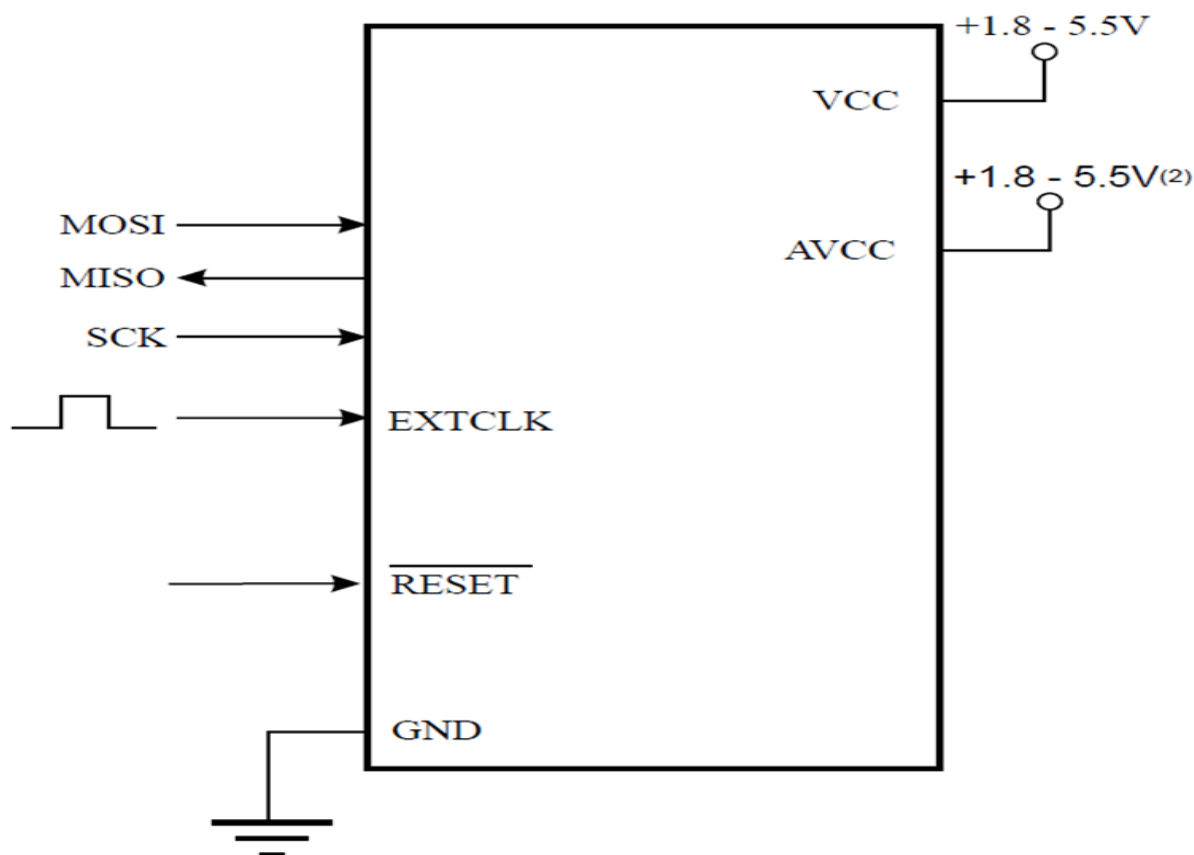
- Las interfaces HVPP o HVSP siempre están habilitadas porque no se pueden deshabilitar inadvertidamente mediante la configuración de un fusible o la acción del usuario.

Las malas noticias ...

- Casi nunca son una opción de programación realista una vez que se suelda un Tiny o Mega en una placa personalizada porque simplemente requieren demasiados pines.

Proveedor de servicios de Internet/JTAG

Además de las interfaces HVPP o HVSP, todos los dispositivos Tiny y Mega (ver Excepciones a continuación) también incluyen una o dos interfaces de programación "estándar": ISP o JTAG. ISP (Programación en serie en circuito) permite que la memoria del programa se re programe en el sistema a través de una interfaz en serie SPI. El siguiente esquema muestra las conexiones ISP requeridas para el ATmega328PB (consulte la sección 33.9 en la hoja de datos):



ATmega328PB ISP Connections



Según los fusibles CKSEL, debe haber un reloj válido para que funcione el ISP.

El ISP se cubre en detalle en la nota de aplicación AVR910 - Programación en el sistema.

Las **buenas noticias** ...

- Estas interfaces estándar solo requieren 3 o 4 pines.

Las malas noticias ...

- Se pueden desactivar fácilmente con la configuración incorrecta de los fusibles.

Interfaces combinadas

Todos los dispositivos Tiny y Mega (ver Excepciones a continuación) incluyen una de las dos siguientes **combinaciones** de interfaces de programación:

- HVPP (o HVSP) e ISP
- HVPP (o HVSP) e ISP y JTAG

Las interfaces ISP y JTAG son las interfaces de programación estándar para los dispositivos Tiny y Mega. Se recomienda incluir un encabezado de programación para una de las dos interfaces en cualquier placa personalizada para permitir una reprogramación conveniente del dispositivo si es necesario. **Si un dispositivo solo tiene una interfaz ISP y se desactiva a través de una configuración de fusible, la única recuperación es a través de su interfaz HVPP o HVSP (que probablemente no sea físicamente posible) . Si un dispositivo tiene una interfaz ISP y JTAG y una de esas dos está deshabilitada por la configuración del fusible, la otra interfaz se puede usar para acceder a la parte si se puede acceder a los pines requeridos.**

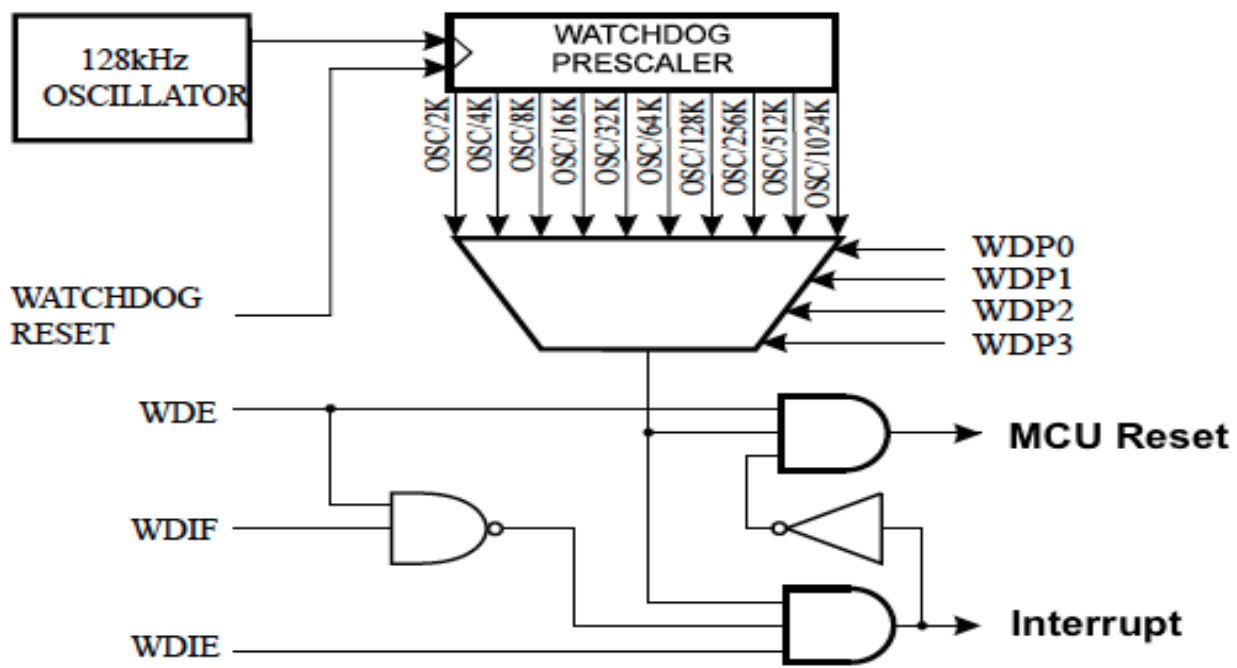
. La interfaz ISP requiere 3 pines y la interfaz JTAG requiere 4 pines, por lo que es más probable que cualquiera de esas interfaces sea más fácil de acceder que las interfaces HVSP o HVPP.

Excepciones

- Los **dispositivos Tiny4/5/9/10/20/40** no tienen una interfaz HVPP o HVSP. Solo tienen una **TPI** (Tiny Programming Interface). Los dispositivos más nuevos, como los dispositivos **Tiny417/817/1617** , solo tienen **UPDI** (Interfaz unificada de programación y depuración). Siempre que tenga acceso a esas interfaces, no hay que preocuparse por "bloquear" estos dispositivos.
- Los dispositivos Xmega incluyen solo una **PDI** (Interfaz de programación y depuración) o una interfaz PDI y JTAG. No puede deshabilitar la interfaz PDI, siempre que tenga acceso al pin PDI que no se puede usar para ninguna otra función y al pin Restablecer, no puede "bloquear" estos dispositivos.

Temporizador de vigilancia AVR®

Los dispositivos AVR® tienen un temporizador de vigilancia mejorado (WDT) que se ejecuta en un oscilador separado del reloj de instrucciones principal. El WDT es esencialmente un contador que se incrementa en función de los ciclos de reloj de un oscilador de 128 kHz en el chip. El WDT fuerza una interrupción o un reinicio del sistema cuando el contador alcanza un valor de tiempo de espera determinado. En el modo de funcionamiento normal, el código de la aplicación debe emitir una instrucción de restablecimiento del temporizador de vigilancia (WDR) para reiniciar el contador antes de que se alcance el valor de tiempo de espera. Si el sistema no reinicia el contador, se emitirá una interrupción o un reinicio del sistema.



Breve resumen de WDT

FUENTE YOUTUBE: <https://youtu.be/E4m6nhWZNzA>

Características de WDT:

- Con reloj desde un oscilador en chip separado.
- Tres modos de funcionamiento:
 - Interrumpir
 - Reinicio de sistema
 - Interrupción y reinicio del sistema
- Período de tiempo de espera seleccionable de 16 milisegundos a 8 segundos.
- Fusible de hardware opcional Watchdog siempre encendido (WDTON) para el modo a prueba de fallas.

Modo de interrupción

En el modo de interrupción, el WDT fuerza una interrupción cuando expira el temporizador. Esta interrupción se puede utilizar para activar el dispositivo desde cualquiera de los modos de suspensión y también como un temporizador general del sistema. Un ejemplo es limitar el tiempo máximo permitido para ciertas operaciones, forzando una interrupción cuando la operación ha durado más de lo esperado. Esto se habilita configurando el bit de modo de interrupción (WDIE) en el registro de control del temporizador de vigilancia (WDTCSR).

Reinicio de sistema

En el modo de reinicio del sistema, el WDT fuerza un reinicio cuando expira el temporizador. Esto generalmente se usa para evitar que el sistema se cuelgue en el caso de un código fuera de control. Esto se habilita configurando el bit de modo de reinicio del sistema (WDE) en el registro de control del temporizador de vigilancia (WDTCSR).

Modo de interrupción y reinicio del sistema

El modo de interrupción y restablecimiento del sistema combina los otros dos modos forzando primero una interrupción y luego cambiando al modo de restablecimiento del sistema. Este modo ofrecerá un apagado seguro al dar tiempo para guardar parámetros críticos antes de reiniciar el sistema. Esto está habilitado cuando tanto WDTIE como WDTE están configurados.

Período de tiempo de espera de WDT

Los bits de preescala del temporizador de vigilancia (WDP[3:0]) del WDTCSR determinan el retraso del WDT cuando el WDT se está ejecutando. Los diferentes valores de preescalado y sus correspondientes tiempos de espera se muestran en la siguiente tabla.

WDP3	WDP2	WDP1	WDP0	Number of WDT Oscillator (Cycles)	Oscillator
0	0	0	0	2K (2048)	16ms
0	0	0	1	4K (4096)	32ms
0	0	1	0	8K (8192)	64ms
0	0	1	1	16K (16384)	0.125s
0	1	0	0	32K (32768)	0.25s
0	1	0	1	64K (65536)	0.5s
0	1	1	0	128K (131072)	1.0s
0	1	1	1	256K (262144)	2.0s
1	0	0	0	512K (524288)	4.0s
1	0	0	1	1024K (1048576)	8.0s



WDTON

El fusible WDTON, si está programado, obligará al WDT a entrar en el modo de reinicio del sistema. Con el fusible programado, el bit WDE y el bit WDIE se bloquean en 1 y 0 respectivamente.

Indicador de restablecimiento del sistema de vigilancia

El bit de indicador de restablecimiento del sistema de vigilancia (WDRF) en el registro de estado de la MCU (MCUSR) se establece si se produce un restablecimiento del sistema de vigilancia. El bit WDRF se borra mediante un reinicio de encendido o escribiendo '0' en él. Para identificar una condición de reinicio, el usuario debe leer y luego reiniciar el WDRF lo antes posible en el programa. Si el registro se borra antes de que ocurra otro restablecimiento, la fuente del restablecimiento se puede encontrar examinando los indicadores de restablecimiento.

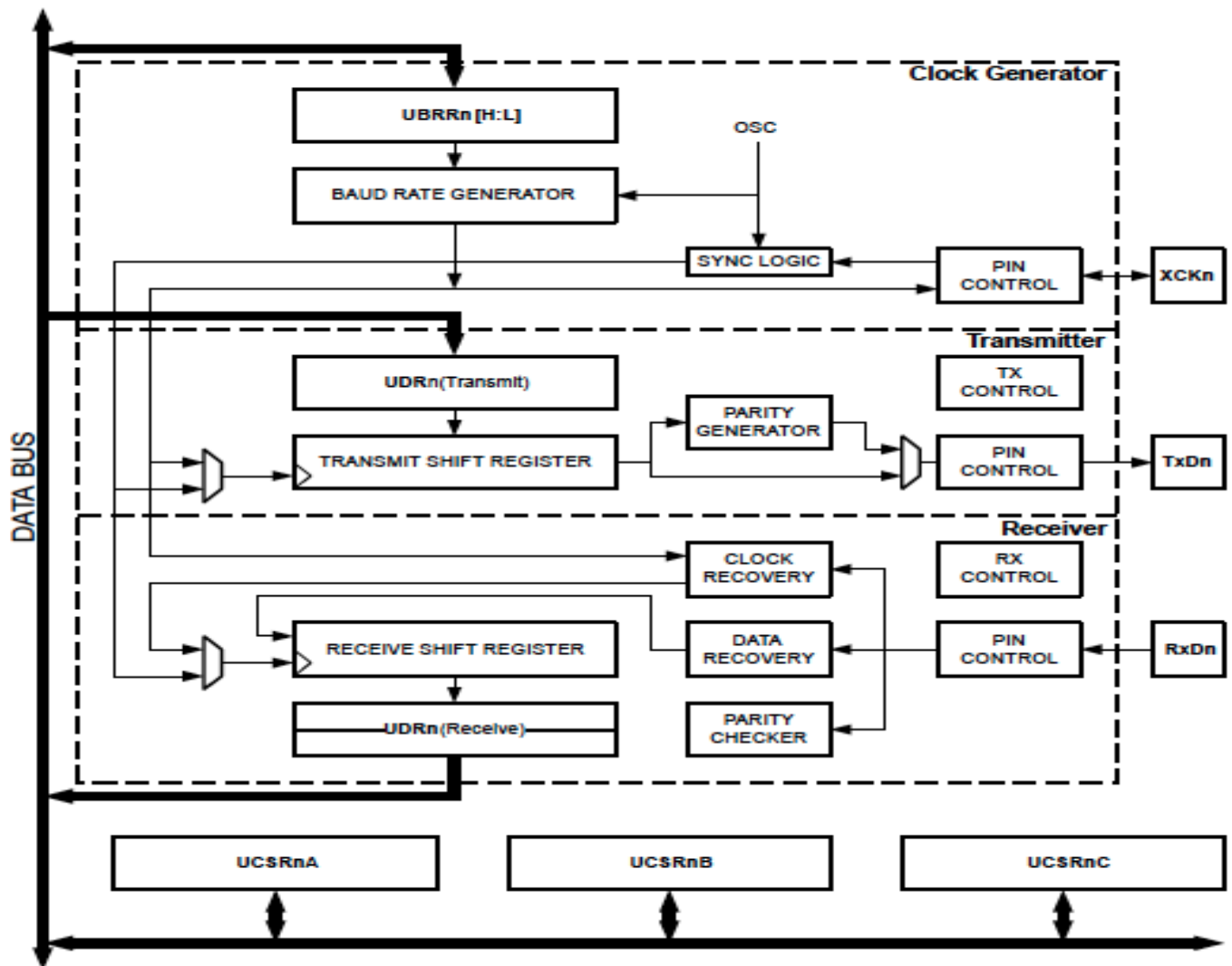
Registro de estado de MCU

Bit	7	6	5	4	3	2	1	0
					WDRF	BORF	EXTRF	PORF
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

AVR® USART Introducción

Los dispositivos AVR® incluyen al menos uno y, a veces, más bloques de receptor y transmisor serie síncrono universal y asíncrono (USART). En el diagrama de bloques del USART, se muestran los registros de E/S accesibles de la Unidad Central de Procesamiento (CPU) y los pines de E/S.

Diagrama de bloques USART



Los cuadros discontinuos en el diagrama de bloques separan las tres partes principales del USART:

- Generador de reloj
- Transmisor
- Receptor

La lógica de **generación** de reloj consiste en la lógica de sincronización para la entrada de reloj externa utilizada por la operación de esclavo síncrono y el generador de velocidad en baudios. El **pin XCKn (reloj de transferencia)** solo se utiliza en el modo de transferencia síncrono.

El **transmisor** consta de un único búfer de escritura, un registro de desplazamiento en serie, un generador de paridad y una lógica de control para manejar diferentes formatos de trama serie. El búfer de escritura permite una transferencia continua de datos sin ningún retraso entre tramas.

El **receptor** es la parte más compleja del módulo USART debido a su reloj y unidades de recuperación de datos. Las unidades de recuperación se utilizan para la recepción asincrónica de datos. Además de las unidades de recuperación, el receptor incluye un comprobador de paridad, lógica de control, un registro de mayús y un búfer de recepción de dos niveles (UDRn). El receptor admite los mismos formatos de trama que el transmisor y puede detectar errores de trama, desbordamiento de datos y errores de paridad.

Breve introducción a USART

Características de USART:

- Operación Full Duplex (Registros de Recepción y Transmisión Serie Independientes)
- Operación asincrónica o síncrona
- Operación síncrona con reloj maestro o esclavo
- Generador de velocidad en baudios de alta resolución
- Admite tramas serie con 5, 6, 7, 8 o 9 bits de datos y 1 o 2 bits de parada
- Generación de paridad impar o par y comprobación de paridad admitida por hardware
- Detección de sobrepaso de datos
- Detección de errores de encuadre
- El filtrado de ruido incluye detección de bits de inicio falso y filtro digital de paso bajo
- Tres interrupciones separadas en TX Complete, TX Data Register Empty y RX Complete
- Modo de comunicación multiprocesador
- Modo de comunicación asíncrono de doble velocidad
- Iniciar la detección de fotogramas

Comunicación en serie

La comunicación en serie es una forma de enviar datos entre dos dispositivos electrónicos utilizando solo dos cables. El USART transmite y recibe datos utilizando el formato estándar de no retorno a cero (NRZ). NRZ se implementa con dos niveles: un estado de salida de voltaje alto (VOH) o marca que representa un bit de datos '1', y un estado de salida de voltaje bajo (VOL) o espacio que representa un bit de datos '0'. NRZ se refiere al hecho de que los bits de datos transmitidos consecutivamente del mismo estado permanecen en el mismo nivel de salida sin volver a un nivel cero o neutro entre cada transmisión de bits.

El estado Idle coloca el pin de salida en una marca (es decir, estado alto). La transmisión puede ocurrir de manera síncrona o asíncrona.

Funcionamiento del reloj síncrono

Las comunicaciones serie síncronas se utilizan normalmente en sistemas con un solo maestro y uno o más esclavos. El dispositivo maestro contiene los circuitos necesarios para la generación de velocidad en baudios y suministra el reloj para todos los dispositivos del sistema. Los dispositivos esclavos pueden aprovechar el reloj maestro eliminando los circuitos internos de generación de relojes. Hay dos líneas de señal en modo síncrono:

- Línea de datos bidireccional
- Línea de reloj

Los esclavos utilizan el reloj externo suministrado por el maestro para mover los datos en serie dentro y fuera de sus respectivos registros de recepción y transmisión. Dado que la línea de datos es bidireccional, la operación síncrona es solo semidúplex. Half-duplex se refiere al hecho de que los dispositivos maestro y esclavo pueden recibir y transmitir datos, pero no ambos simultáneamente. El USART puede funcionar como un dispositivo maestro o esclavo. Normalmente, los bits start y stop no son necesarios para las transmisiones síncronas

Cuando se utiliza el modo síncrono (UMSEL = 1), el pin **XCKn** se utilizará como entrada de reloj (esclavo) o salida de reloj (maestro). La dependencia entre los bordes del reloj y el muestreo de datos o el cambio de datos es la misma. El principio básico es que la entrada de datos (pin **RxDn**) se muestrea en el borde de reloj **XCKn** opuesto del borde, se cambia la salida de datos (pin **TxDn**).

Recepción asíncrona de datos

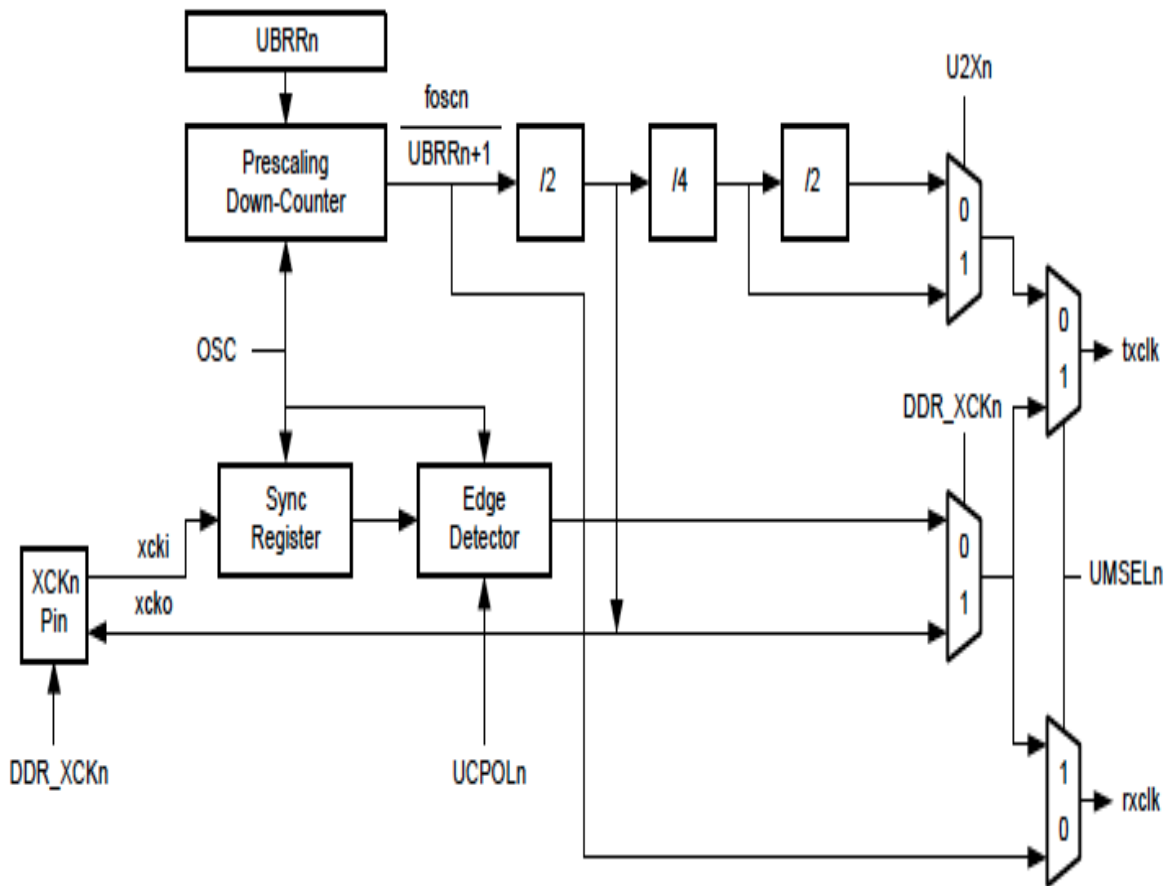
En las comunicaciones asíncronas se utilizan dos pines. Uno es el Transmit (**TxDn**) y otro es el Receive (**RxDn**). El **TxDn** de un dispositivo está conectado al **RxDn** del segundo dispositivo. Cada transmisión de caracteres consta de un bit Start seguido de cinco a nueve bits de datos y siempre termina con uno o más bits Stop. El bit Start es siempre un espacio y los bits Stop son siempre marcas. Cada bit transmitido persiste durante un período de $1/(\text{Velocidad en baudios})$. Se utiliza un generador de velocidad de baudios dedicado en chip para derivar frecuencias de velocidad en baudios estándar del oscilador del sistema.

Generación de relojes

La lógica de generación de reloj genera el reloj base para el transmisor y el receptor. El USART admite cuatro modos de funcionamiento del reloj:

- Asíncrono normal
- Doble velocidad asíncrona

Diagrama de bloques de reloj



Descripción de la señal:

- txclk Reloj transmisor (señal interna).
- rxclk Reloj base del receptor (señal interna).
- Entrada xcki desde el pin XCK (señal interna). Se utiliza para la operación esclava síncrona.
- Salida xcko Clock al pin XCK (señal interna). Se utiliza para la operación maestra síncrona.
- Frecuencia de reloj del sistema OSC.

Generador de velocidad en baudios

La generación de reloj interno se utiliza para los modos de funcionamiento maestro asíncrono y síncrono. El REGISTRO DE VELOCIDAD EN BAUDIOS USART (**UBRRn**) y el contador descendente conectado a él, funcionan como un preescalador programable o generador de velocidad en baudios. El contador descendente, que se ejecuta en el reloj del sistema (fosc), se carga con el valor **UBRRn** cada vez que el contador ha contado hasta cero o cuando se escribe el registro **UBRRnL**. Se genera un reloj cada vez que

el contador llega a cero. Este reloj es la salida del reloj del generador de velocidad en baudios ($= f_{osc}/(UBRRn+1)$).

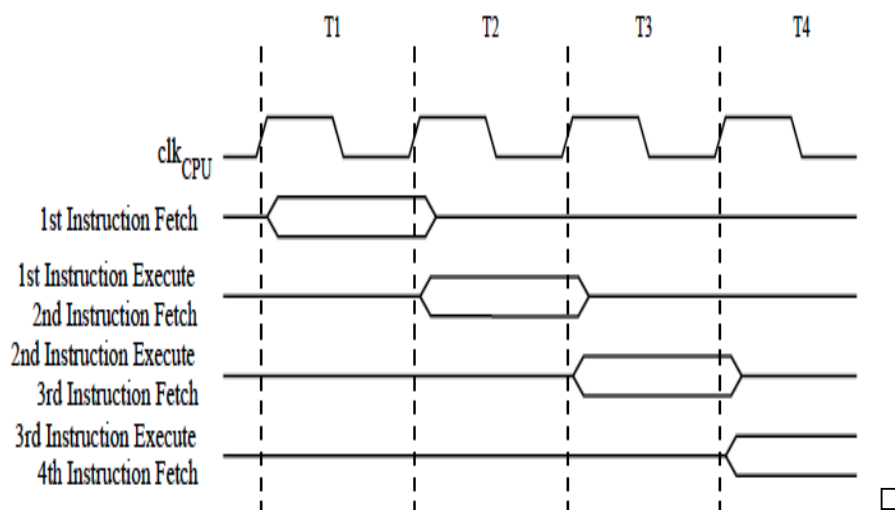
La siguiente tabla contiene ecuaciones para calcular la velocidad en baudios (en bits por segundo) y para calcular el valor **UBRRn** para cada modo de operación utilizando una fuente de reloj generada internamente.

Operating Mode	Equation for Calculating Baud Rate ⁽¹⁾	Equation for Calculating UBRRn Value
Asynchronous Normal mode (U2X = 0)	$BAUD = \frac{f_{osc}}{16(UBRRn + 1)}$	$UBRRn = \frac{f_{osc}}{16BAUD} + -1$
Asynchronous Double Speed mode (U2X = 1)	$BAUD = \frac{f_{osc}}{8(UBRRn + 1)}$	$UBRRn = \frac{f_{osc}}{8BAUD} + -1$
Synchronous Master mode	$BAUD = \frac{f_{osc}}{2(UBRRn + 1)}$	$UBRRn = \frac{f_{osc}}{2BAUD} + -1$

Temporización de instrucciones AVR® de 8 bits

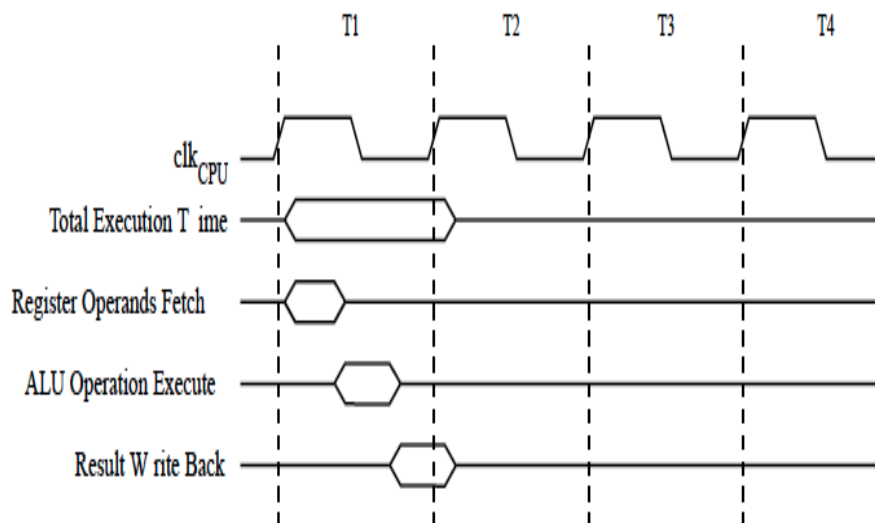
La unidad central de procesamiento (CPU) AVR es impulsada por el reloj `clkCPU` de la CPU, generado directamente desde la fuente de reloj seleccionada para el chip. No se utiliza ninguna división de reloj interna. La arquitectura de Harvard y el concepto de archivo de registro de acceso rápido permiten obtener y ejecutar instrucciones en paralelo. Este es el concepto básico de canalización para obtener hasta 1 MIPS por MHz con los resultados únicos correspondientes para funciones por costo, funciones por relojes y funciones por unidad de potencia.

Obtención de instrucciones en paralelo y ejecución de instrucciones



En un solo ciclo de reloj, se ejecuta una operación de unidad lógica aritmética (ALU) utilizando dos operandos de registro y el resultado se almacena nuevamente en el registro de destino.

Operación ALU de ciclo único



Modos de funcionamiento de AVR ADC

Microchip Studio es un entorno de desarrollo integrado (IDE) para desarrollar y depurar aplicaciones de microcontroladores AVR® y SAM. Combina todas las excelentes características y funcionalidades de Atmel Studio en la cartera de herramientas de desarrollo bien respaldada de Microchip para brindarle un entorno sencillo y fácil de usar para escribir, construir y depurar sus aplicaciones escritas en C/C++ o código ensamblador. Microchip Studio también puede importar sus bocetos de Arduino® como proyectos de C++ para brindarle una ruta de transición simple del makerspace al mercado.

Puede utilizar Microchip Studio con los depuradores, programadores y kits de desarrollo compatibles con dispositivos AVR y SAM. Amplíe su entorno de desarrollo con Microchip Gallery, una tienda de aplicaciones en línea para los complementos de Microchip Studio desarrollados por Microchip, así como por proveedores de software integrado y herramientas de terceros.

Aunque viene con un nombre y una apariencia nuevos, aún podrá usar cualquier documentación y videos existentes sobre Atmel Studio para aprender a usar Microchip Studio.

[Descargar Microchip Studio](#)



Licencias MPLAB® XC8 PRO ahora repletas de rendimiento para MCU AVR

Microchip Studio viene con el compilador MPLAB XC8 instalado y listo para usar. La última versión de la licencia MPLAB XC8 PRO incluye optimizaciones que compiten con las de los compiladores más caros del mercado para reducir el tamaño del código y aumentar la eficiencia. También está disponible una licencia de seguridad funcional certificada por TÜV SÜD.

- [Pruebe una licencia de evaluación MPLAB XC8 PRO de 60 días](#)
- [Obtenga más información sobre las licencias MPLAB XC8](#)
- [Más información sobre las licencias de seguridad funcional MPLAB XC8](#)

Microchip Studio es un entorno de desarrollo integrado (IDE) para desarrollar y depurar aplicaciones de microcontroladores AVR® y SAM. Combina todas las excelentes características y funcionalidades de Atmel Studio en la cartera de herramientas de desarrollo bien respaldada de Microchip para brindarle un entorno sencillo y fácil de usar para escribir, construir y depurar sus aplicaciones escritas en C/C++ o código ensamblador. Microchip Studio también puede importar sus bocetos de Arduino® como proyectos de C++ para brindarle una ruta de transición simple del makerspace al mercado.

Puede utilizar Microchip Studio con los depuradores, programadores y kits de desarrollo compatibles con dispositivos AVR y SAM. Amplíe su entorno de desarrollo con Microchip Gallery, una tienda de aplicaciones en línea para los complementos de Microchip Studio desarrollados por Microchip, así como por proveedores de software integrado y herramientas de terceros.

Aunque viene con un nombre y una apariencia nuevos, aún podrá usar cualquier documentación y videos existentes sobre Atmel Studio para aprender a usar Microchip Studio.

[Consulte este enlace para obtener información sobre nuestros avisos de seguridad .](#)

[Descargar Microchip Studio](#)



Licencias MPLAB® XC8 PRO ahora repletas de rendimiento para MCU AVR

Microchip Studio viene con el compilador MPLAB XC8 instalado y listo para usar. La última versión de la licencia MPLAB XC8 PRO incluye optimizaciones que compiten con las de los compiladores más caros del mercado para reducir el tamaño del código y aumentar la eficiencia. También está disponible una licencia de seguridad funcional certificada por TÜV SÜD.

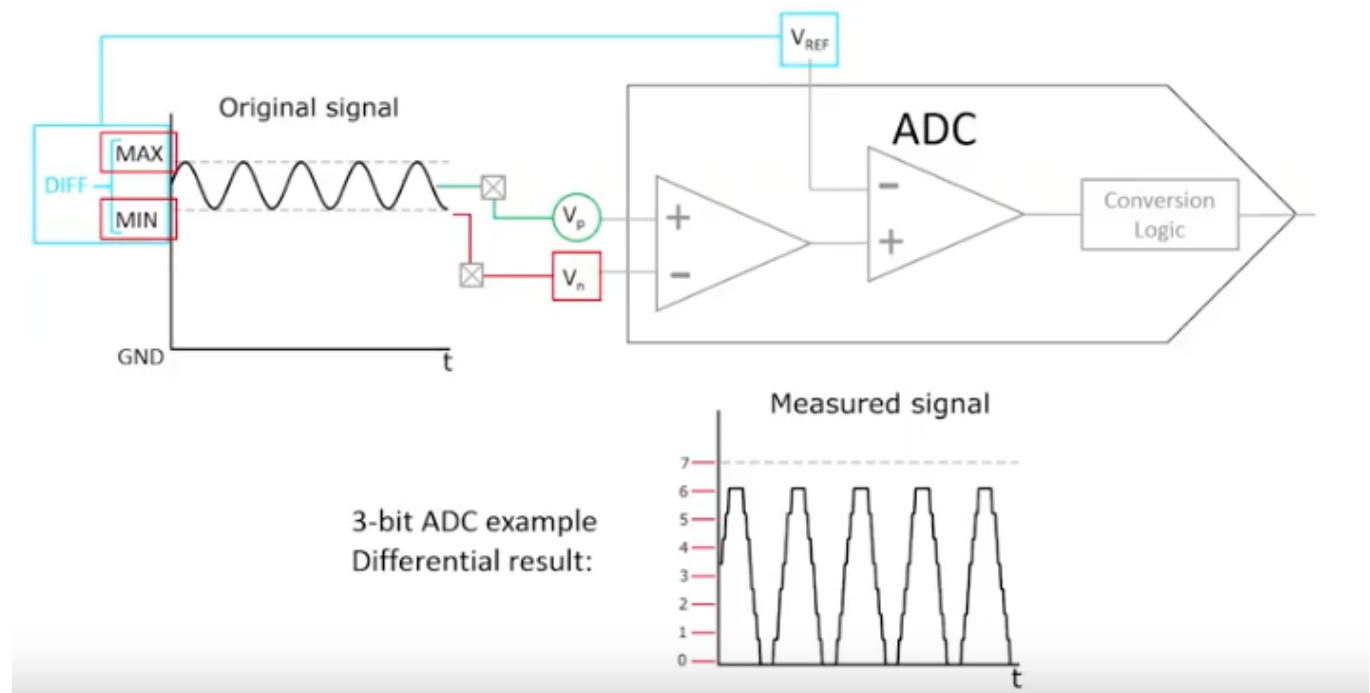
- [Pruebe una licencia de evaluación MPLAB XC8 PRO de 60 días](#)
- [Obtenga más información sobre las licencias MPLAB XC8](#)
- [Más información sobre las licencias de seguridad funcional MPLAB XC8](#)

Modo diferencial AVR® ADC

Un convertidor analógico a digital (ADC) diferencial mide la diferencia de voltaje entre dos señales. Un ADC típicamente mide el voltaje entre la señal y la tierra, pero en el modo diferencial, el pin de tierra está realmente conectado a otra parte del circuito, por lo que el ADC puede medir la diferencia entre las dos señales. Esto se usa a menudo para medir una señal pequeña con un desplazamiento grande. El uso de una entrada diferencial permite que el ADC mida una porción más pequeña de la señal.

Configuración diferencial

Las entradas diferenciales se ejecutan a través de un amplificador de ganancia para aumentar el tamaño de la señal al convertidor. Algunos dispositivos AVR® tienen pasadores con ganancia ajustable. Cuando se utilizan canales de ganancia diferencial, se deben tener en cuenta ciertos aspectos de la conversión. Los canales diferenciales no deben utilizarse con un voltaje de referencia analógico (AREF) inferior a 2 V. En los dispositivos AVR, las conversiones diferenciales se sincronizan con el reloj interno CKADC2 igual a la mitad del reloj de ADC. Esta sincronización se realiza automáticamente por la interfaz ADC de tal forma que la muestra y retención ocurre en una fase específica del reloj CKADC2.



Tiempo de mediciones críticas

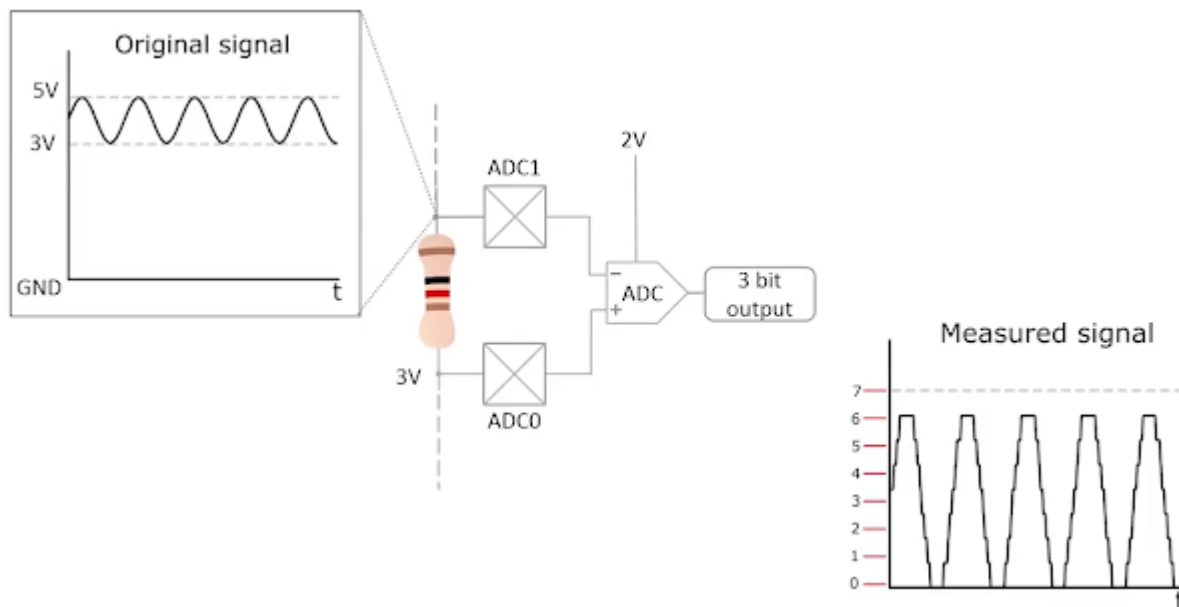
En un dispositivo AVR típico con modo diferencial, una conversión que inicie (es decir, todas las conversiones individuales y la primera conversión de ejecución gratuita) cuando la señal de reloj CKADC2 es baja tomará la misma cantidad de tiempo que una conversión de terminación única (13 ADC reloj ciclos desde el siguiente ciclo de reloj preescalado). Una conversión que inicie cuando la señal de reloj CKADC2 es alta tomará 14 ciclos de reloj ADC debido al mecanismo de sincronización.

En el modo de ejecución gratuita, se inicia una nueva conversión inmediatamente después de que se complete la conversión anterior, y dado que CKADC2 es alta en este momento, todas las conversiones de ejecución automática iniciadas automáticamente (es decir, todas menos la primera) tomarán 14 ciclos de reloj ADC.

Si se utilizan canales de ganancia diferencial y las conversiones se inician con el disparo automático, el ADC se debe desconectar entre las conversiones. Cuando se utiliza el disparo automático, el preescalador ADC se reinicia antes de que se inicie la conversión. Como la etapa de ganancia depende de un reloj de ADC estable antes de la conversión, esta conversión no será válida. Al deshabilitar y luego volver a habilitar el ADC entre cada conversión (estableciendo el bit ADEN en ADCSRA para luego en 1), solo se realizan conversiones extendidas. El resultado de las conversiones ampliadas será válido.

Aplicación típica

Leer el voltaje a través de una resistencia con un desplazamiento constante de tres voltios es un ejemplo simple donde una entrada diferencial puede medir la señal por encima del desplazamiento de CC.



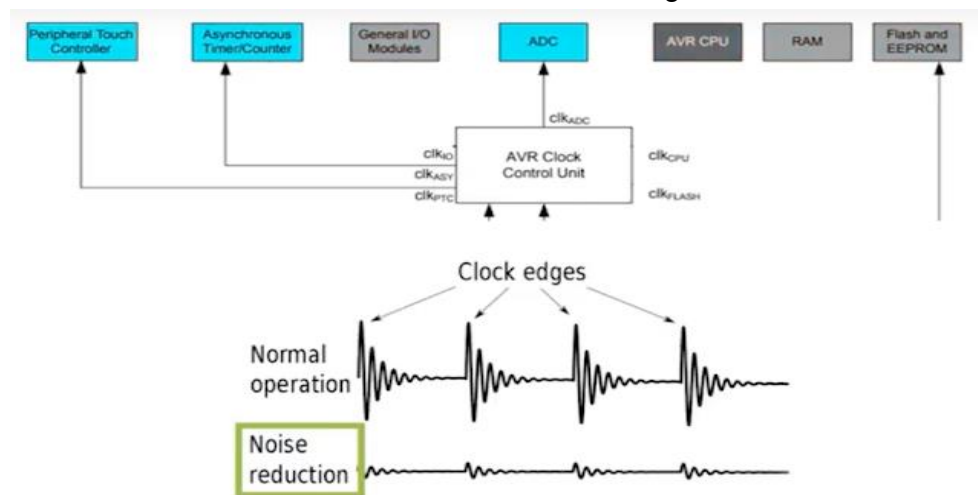
Fuente: AVR® ADC Differential Mode

Modo de reducción de ruido AVR ADC

Los dispositivos AVR® tienen un modo de reducción de ruido del convertidor analógico a digital (ADC), que detiene la CPU y todos los módulos de E / S, excepto el temporizador asincrónico, PTC y ADC, para minimizar el ruido de conmutación durante las conversiones ADC. Se usa cuando se requiere una medición de ADC de alta resolución. Las mediciones de ADC luego se implementan cuando el núcleo se pone a dormir.

Entrar / Salir de Reducción de ruido ADC

La instrucción SLEEP hace que la MCU ingrese al modo de reducción de ruido ADC, deteniendo la CPU pero permitiendo que el ADC, las interrupciones externas, la vigilancia de la dirección de la interfaz serie de dos hilos, Timer1 y Watchdog continúen funcionando (si está habilitado). Este modo de suspensión básicamente detiene clkI / O, clkCPU y clkFLASH, mientras permite que los otros relojes se ejecuten. Esto mejora el ambiente de ruido para el ADC, permitiendo mediciones de mayor resolución. Si el ADC está habilitado, una conversión se inicia automáticamente cuando se ingresa este modo.



Además de la interrupción completa de conversión de ADC, solo estos eventos pueden activar la MCU del modo de reducción de ruido de ADC:

Restablecer externo

Sistema de vigilancia Reset

Interrupción del perro guardián

Restablecer Brown-out

Coincidencia de dirección de la interfaz serie de dos hilos

Interrupción de temporizador / contador

Interrupción lista SPM / EEPROM

Nivel externo de interrupción en INT

Interrupción de cambio de pin

Nota: El temporizador / contador solo se ejecuta en modo asíncrono.

Referencia de voltaje AVR® ADC

La referencia analógica (AREF) es el voltaje de referencia del convertidor analógico al digital (ADC) en chip en dispositivos AVR. El voltaje de referencia para el ADC, VREF, indica el rango de voltaje de la conversión ADC. Los canales de un solo extremo que superan VREF dan como resultado un valor de conversión máximo. VREF se puede medir en el pin AREF con un voltímetro de alta impedancia.®

<https://youtu.be/fN3J6wxijrE>

Opciones de VREF

VREF se puede seleccionar como AVCC, referencia interna de 1,1 V o pin AREF externo.

- AVCC es el voltaje conectado al pin AVCC que está conectado internamente al ADC a través de un interruptor pasivo.
- La referencia interna de 1,1 V se genera a partir de la referencia de banda prohibida interna (V_{BG}) a través de un amplificador interno.
- El pin AREF externo está conectado directamente al ADC, y el voltaje de referencia se puede hacer más inmune al ruido conectando un condensador entre el pin AREF y la tierra.

Si una fuente de voltaje fijo está conectada al pin AREF externo, la aplicación no puede usar las otras opciones de voltaje de referencia en la aplicación, ya que se cortocircuitarán al voltaje externo.

Si no se aplica voltaje externo al pin AREF, puede cambiar entre AVCC y 1.1 V como selección de referencia. El primer resultado de conversión de ADC después de cambiar la fuente de voltaje de referencia puede ser inexacto, y se recomienda descartar este resultado.

Si se utilizan canales diferenciales, la referencia seleccionada no debe estar más cerca de AVCC de lo indicado en las "Características ADC de las características eléctricas" en la hoja de datos del dispositivo

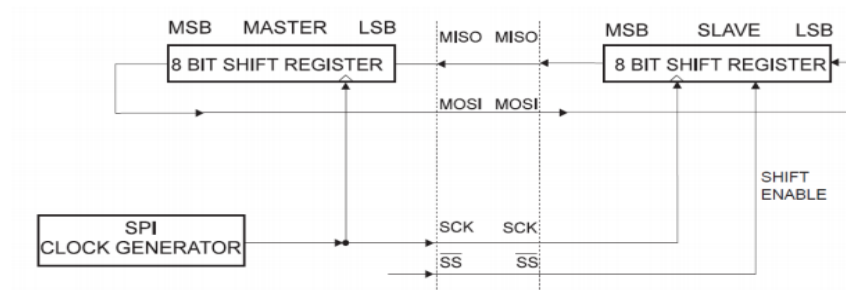
Interfaz periférica serie AVR® (SPI)

El protocolo de interfaz periférica serie (SPI) en dispositivos AVR® permitirá que su microcontrolador AVR se comunique con varios otros dispositivos al mismo tiempo. Utilice el bus SPI para comunicarse entre un dispositivo maestro y uno o varios dispositivos esclavos. SPI utiliza las líneas Master In Slave Out (MISO) y Master Out Slave In (MOSI) para comunicarse entre dispositivos, el reloj serie (SCK) para mantener un reloj coherente entre dispositivos y la línea Slave Select (SS) para elegir qué dispositivo periférico se está comunicando con el dispositivo maestro.

https://youtu.be/9DLxM_GwT0A

Sistema SPI

El sistema consta de dos registros de turnos y un generador de reloj maestro. El SPI Master inicia el ciclo de comunicación al bajar el pin SS del Esclavo deseado. El maestro y el esclavo preparan los datos que se enviarán en sus respectivos registros de turnos, y el maestro genera los pulsos de reloj necesarios en la línea SCK para intercambiar datos. Los datos siempre se cambian de Maestro a Esclavo en la línea MOSI, y de Esclavo a Maestro en la línea MISO. Después de cada paquete de datos, el maestro sincronizará el esclavo tirando de la línea SS en alto.



Modo maestro

Cuando se configura como Master, la interfaz SPI no tiene control automático de la línea SS. Esto debe ser manejado por el software del usuario antes de que pueda iniciarse la comunicación. Una vez hecho esto, al escribir un byte en el registro de datos SPI se inicia el generador de reloj SPI y el hardware cambia los ocho bits al esclavo. Después de cambiar un byte, el generador de reloj SPI se detiene, estableciendo el final de la bandera de transmisión (SPIF). Si se establece el bit SPI Interrupt Enable (SPIE) en el registro SPCR, se solicita una interrupción. El maestro puede continuar desplazando el siguiente byte escribiéndolo en SPDR o señalar el final del paquete tirando alto de la línea SS de Selección de esclavo. El último byte entrante se mantendrá en el Registro de búfer para su uso posterior.

Modo esclavo

Cuando se configura como slave, la interfaz SPI permanecerá en reposo con MISO tri-declarado siempre que el pin SS esté alto. En este estado, el software puede actualizar el contenido del registro de datos SPI, SPDR, pero los datos no se desplazarán por los pulsos de reloj entrantes en el pin SCK hasta que el pin SS se reduzca. Como un byte se ha desplazado por completo, se establece el final de la bandera de transmisión (SPIF). Si se establece el bit de habilitación de interrupciones SPI (SPIE) en el registro SPCR, se solicita una interrupción.

Controlador táctil periférico AVR (PTC)

El controlador táctil periférico (PTC), en algunos AVR® dispositivos, se utiliza para aplicaciones táctiles capacitivas. El PTC adquiere señales para detectar un toque en los sensores capacitivos. El sensor táctil capacitivo externo generalmente se forma en una PCB, y los electrodos del sensor se conectan al extremo frontal analógico del PTC a través de los pines de E / S en el dispositivo. El PTC admite sensores de capacitancia propia y mutua.

<https://youtu.be/ZLI7yDypew8>

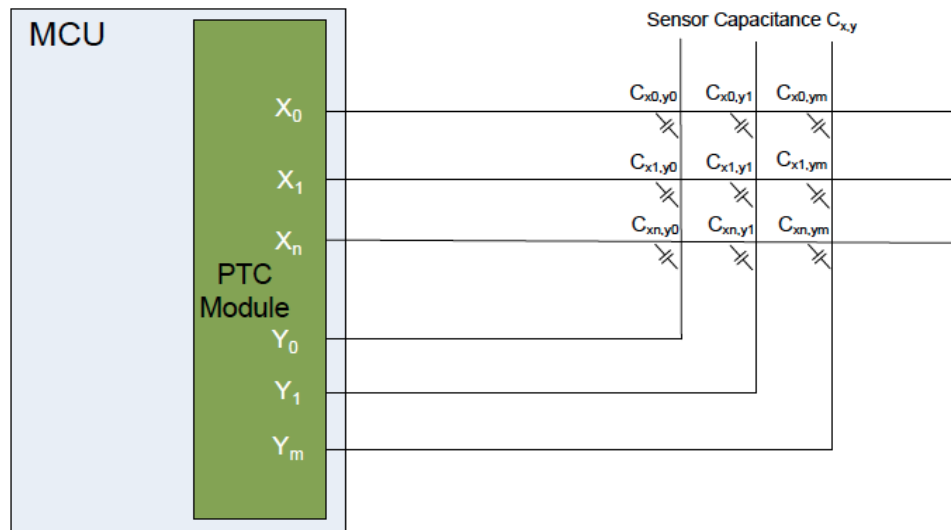
Conexiones de E/S.

Las líneas de E/S utilizadas para las líneas X analógicas y las líneas Y deben conectarse a electrodos de sensor táctil capacitivo externo. Los componentes externos no son necesarios para el funcionamiento normal. Sin embargo, para mejorar el rendimiento de la compatibilidad electromagnética (EMC), se puede utilizar una resistencia en serie de 1k Ω o más en líneas X e Y.

Name	Type	Description
X[n:0]	Digital	X-line (Output)
Y[m:0]	Analog	Y-line (Input/Output)

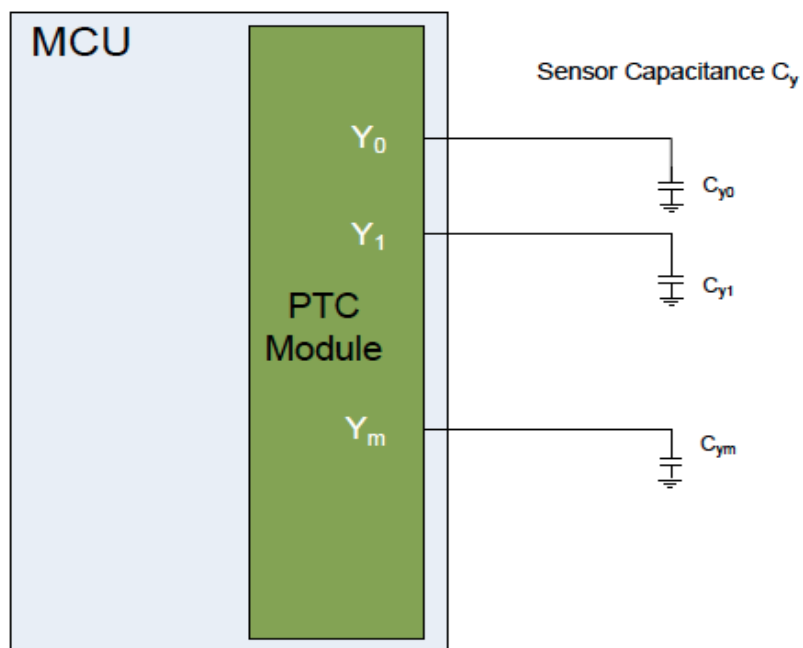
Modo de capacitancia mutua

En el modo de capacitancia mutua, la detección se realiza utilizando matrices táctiles capacitivas en varias configuraciones X-Y, incluidas las rejillas de sensores de óxido de indio y estaño (ITO). El PTC requiere un pin por línea X y un pin por línea Y. Se forma un sensor de capacitancia mutua entre las dos líneas de E/S: un electrodo X para transmitir y un electrodo Y para detectar. La capacitancia mutua entre el electrodo X e Y es medida por el PTC.



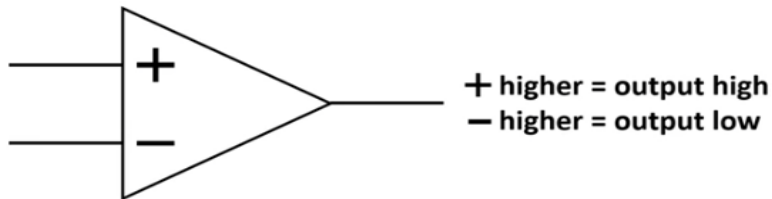
Modo de autocapacitancia

En el modo de autocapacitancia, el PTC solo requiere un pin (línea Y) para cada sensor táctil. Un sensor de autocapacitancia está conectado a un solo pin en el PTC a través del electrodo Y para detectar la señal. La capacitancia del electrodo de detección es medida por el PTC.



Comparador interno AVR

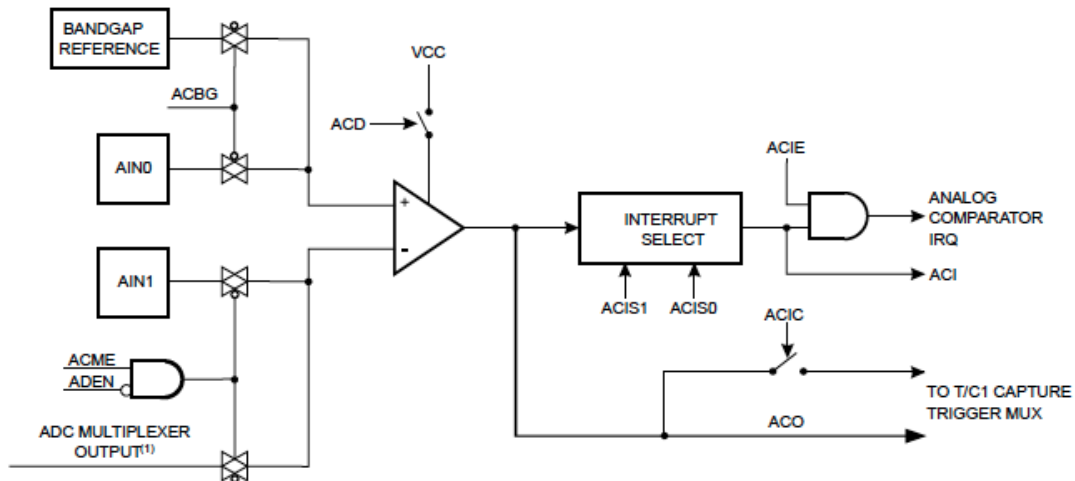
Muchos dispositivos AVR tienen un periférico comparador analógico interno que compara los valores de entrada en el pin positivo **AIN0** y el pin negativo **AIN1**. Cuando el voltaje en el pin positivo AIN0 es mayor que el voltaje en el pin negativo AIN1, se establece la salida del comparador analógico, **ACO**.



Descripción general del comparador

https://youtu.be/wB_4bGv9sYc

Configuración del comparador



Un diagrama de bloques del comparador y su lógica circundante.

Interrupciones del comparador

La salida del comparador se puede configurar para activar la función de captura de entrada Timer/Counter1. Además, el comparador puede activar una interrupción separada, exclusiva del comparador analógico. El usuario puede seleccionar La activación de interrupciones en la salida del comparador sube, baja o alterna.

Opciones del pin de entrada negativo del comparador

Es posible seleccionar cualquiera de los pines ADC[7:0] para reemplazar la entrada negativa al comparador analógico. El multiplexor ADC se utiliza para seleccionar esta entrada y, en consecuencia, el ADC debe estar desactivado para utilizar esta función.

Si el multiplexor de comparador analógico enable bit en el registro de control y estado de ADC B ADCSRB. ACME es '1' y el ADC está desactivado ADCSRA. ADEN=0, a continuación, los tres bits de selección de canal analógico menos significativos en el registro de selección de multiplexores ADC ADMUX. MUX[2:0] seleccione el pin de entrada para reemplazar la entrada negativa al comparador analógico.

Cuando ADCSRB. ACME=0 o ADCSRA. ADEN=1, AIN1 se aplica a la entrada negativa del Comparador Analógico.

ACME	ADEN	MUX[2:0]	Analog Comparator Negative Input
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

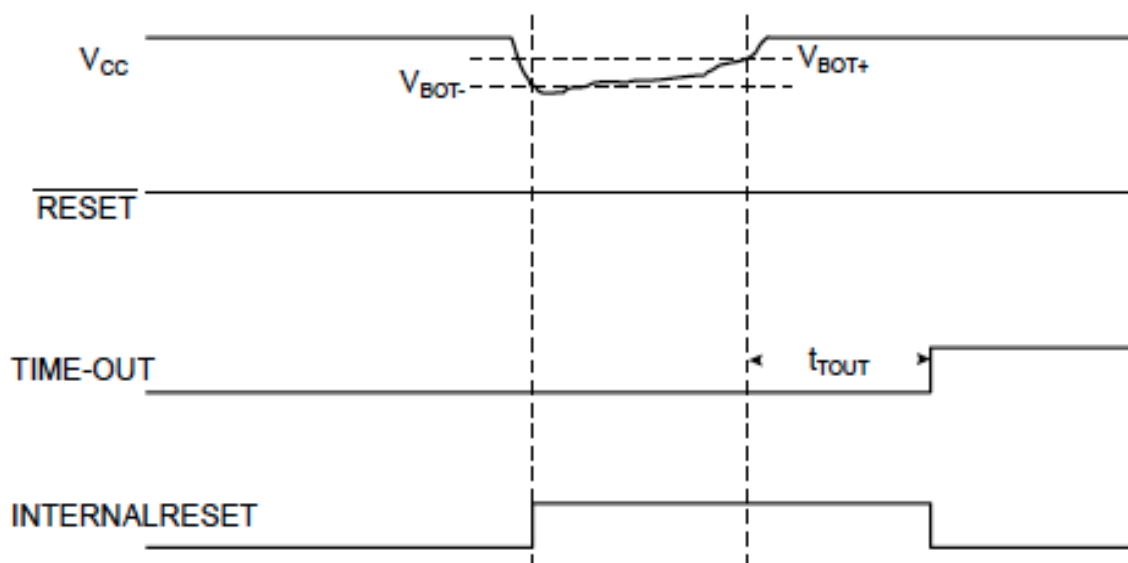
Detección de apagado AVR

Mucho **AVR**® los dispositivos tienen un circuito de detección de salida marrón (DBO) en el chip para monitorear el nivel de voltaje de funcionamiento (VCC) durante la operación. Al comparar el VCC con un nivel de disparo fijo, puede determinar si el dispositivo debe ponerse en modo de reinicio para evitar un funcionamiento errático.

<https://youtu.be/G90Xd2ssxog>

Operación BOD

Brown-out Reset During Operation



El nivel de disparo tiene una histéresis para garantizar una DBO sin picos. Cuando la DBO está habilitada y VCC disminuye a un valor por debajo del nivel de activación (V_{BOT-}), el Brown-out Reset se activa inmediatamente. Cuando VCC aumenta por encima del nivel de activación (V_{BOT+}), el contador de retardo inicia el MCU después del período de tiempo de espera $t_{PRESUMIR}$ ha caducado. El circuito BOD solo detectará una caída en VCC si el voltaje permanece por debajo del nivel de disparo durante más tiempo que el ancho de pulso mínimo Detección de brown-out (t_{Dbo}) especificado en la hoja de datos del dispositivo.

Configuración del fusible BOD

El nivel de disparo para la DBO puede ser seleccionado por los fusibles BODLEVEL cuando se programa el dispositivo. Esta configuración no se puede cambiar mientras se ejecuta el software de la aplicación.

BODLEVEL [2:0] Fuses	Min. V _{BOT}	Typ. V _{BOT}	Max V _{BOT}	Units
111	BOD Disabled			
110	1.7	1.8	2.0	V
101	2.5	2.7	2.9	
100	4.1	4.3	4.5	
011 - 000	Reserved			

Desactivación de BOD

Cuando el detector de apagado (DBO) está habilitado por fusibles BODLEVEL, el DBO monitorea activamente el voltaje de la fuente de alimentación incluso durante un período de suspensión. Para ahorrar energía, es posible desactivar la DBO mediante software para algunos de los modos de suspensión. El consumo de energía del modo de suspensión estará entonces al mismo nivel que cuando la DBO se desactiva globalmente por fusibles.

Si la DBO está deshabilitada en el software, la función DBO se desactiva inmediatamente después de entrar en el modo de suspensión. Al despertar del modo de suspensión, la DBO se vuelve a habilitar automáticamente. Esto garantiza un funcionamiento seguro en caso de que el nivel de VCC haya disminuido durante el período de sueño.

Cuando la DBO se ha desactivado, el tiempo de activación del modo de suspensión será de aproximadamente 60 μ s para garantizar que la DBO funcione correctamente antes de que la MCU continúe ejecutando el código.

La desactivación de DBO está controlada por el bit de suspensión de DBO en el registro de control de MCU MCUCR. DBO. Establecer este bit en '1' apaga la DBO en los modos de suspensión relevantes, mientras que un '0' en este bit mantiene la DBO activa. La configuración predeterminada, MCUCR. DBO = 0, mantiene activa la DBO.

Interrupciones AVR®

AVR® los dispositivos proporcionan varias fuentes de interrupciones diferentes, [incluidas las interrupciones internas y externas](#). Las interrupciones pueden impedir que el programa principal se ejecute para realizar una rutina de servicio de interrupciones (ISR) separada. Cuando se completa el ISR, el control del programa se devuelve al programa principal en la instrucción que se interrumpió.

Cada una de estas interrupciones tiene un vector de programa separado en el espacio de memoria del programa. A todas las interrupciones se les asignan bits de habilitación individuales que deben escribirse en una lógica junto con el bit de habilitación de interrupción global en el Registro de estado para habilitar la

interrupción. Las direcciones más bajas en el espacio de memoria del programa se definen de forma predeterminada como los vectores de restablecimiento e interrupción. Han determinado los niveles de prioridad; cuanto menor sea la dirección, mayor será el nivel de prioridad. RESET tiene la prioridad más alta, y la siguiente es la solicitud de interrupción externa 0 (INT0).

Tabla de vectores de interrupción para ATmega324PB:

Vector No	Program Address ⁽²⁾	Source	Interrupts definition
1	0x0000 ⁽¹⁾	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	INT2	External Interrupt Request 2
5	0x0008	PCINT0	Pin Change Interrupt Request 0
6	0x000A	PCINT1	Pin Change Interrupt Request 1
7	0x000C	PCINT2	Pin Change Interrupt Request 2
8	0x000E	PCINT3	Pin Change Interrupt Request 3
9	0x0010	WDT	Watchdog Time-out Interrupt
10	0x0012	TIMER2_COMPA	Timer/Counter2 Compare Match A
11	0x0014	TIMER2_COMPB	Timer/Counter2 Compare Match B
12	0x0016	TIMER2_OVF	Timer/Counter2 Overflow
13	0x0018	TIMER1_CAPT	Timer/Counter1 Capture Event
14	0x001A	TIMER1_COMPA	Timer/Counter1 Compare Match A
15	0x001C	TIMER1_COMPB	Timer/Counter1 Compare Match B
16	0x001E	TIMER1_OVF	Timer/Counter1 Overflow
17	0x0020	TIMER0_COMPA	Timer/Counter0 Compare Match A
18	0x0022	TIMER0_COMPB	Timer/Counter0 Compare Match B
19	0x0024	TIMER0_OVF	Timer/Counter0 Overflow
20	0x0026	SPI0_STC	SPI0 Serial Transfer Complete
21	0x0028	USART0_RX	USART0 Rx Complete
22	0x002A	USART0_UDRE	USART0, Data Register Empty

Los vectores de interrupción se pueden mover al inicio de la sección Flash de arranque configurando el bit IVSEL en el registro de control MCU MCUCR. El vector de restablecimiento también se puede mover al inicio de la sección Flash de arranque programando el fusible BOOTRST.

<https://youtu.be/onfpksEIXzg>

Cómo funciona

Cuando se produce una interrupción, el bit I de habilitación de interrupción global se borra y todas las interrupciones se deshabilitan. El vector de interrupción dirige el control del programa al ISR o ejecución adecuados. Ese ISR puede escribir lógica uno en el I-bit para habilitar interrupciones anidadas. Todas las interrupciones habilitadas pueden interrumpir la rutina de interrupciones actual. Cuando se completa el ISR

y se ejecuta el comando return (RETI) desde el ISR, el I-bit global se establece automáticamente en 'ON' y la ejecución del programa regresa al programa principal en la instrucción que se interrumpió.

Tiempo de respuesta de interrupción

La respuesta de ejecución de interrupciones para todas las interrupciones AVR habilitadas es de cuatro ciclos de reloj como mínimo. Después de cuatro ciclos de reloj, se ejecuta la dirección vectorial del programa para la rutina real de manejo de interrupciones. Durante este período de ciclo de cuatro relojes, el contador de programas se inserta en la pila. El vector es normalmente un salto a la rutina de interrupción, y este salto toma tres ciclos de reloj. Si se produce una interrupción durante la ejecución de una instrucción de varios ciclos, esta instrucción se completa antes de que se sirva la interrupción.

Si se produce una interrupción cuando la MCU está en modo de suspensión, el tiempo de respuesta de ejecución de interrupciones se incrementa en cuatro ciclos de reloj. Este aumento se suma al tiempo de arranque del modo de suspensión seleccionado. Un retorno de una rutina de manejo de interrupciones toma cuatro ciclos de reloj. Durante estos cuatro ciclos de reloj, el contador de programa (dos bytes) se vuelve a sacar de la pila, el puntero de la pila se incrementa en dos y se establece el bit I en SREG

Interrupciones externas AVR

AVR® los dispositivos tienen interrupciones externas que pueden despertar un dispositivo del modo de suspensión en función de una señal de borde ascendente o descendente en un pin de E/S o un cambio en el nivel de voltaje digital en un pin de E/S. El dispositivo puede procesar una aplicación basada en la fuente de interrupción y luego volver a entrar en reposo. El dispositivo tiene varios pines de interrupción para múltiples fuentes de interrupción.

<https://youtu.be/121d4HVkADs>

Interrupciones externas

Las interrupciones externas son activadas por el pin INT o cualquiera de los pines PCINT. Si está habilitado, las interrupciones se activan incluso si los pines INT o PCINT están configurados como salidas. Esta característica proporciona una forma de generar una interrupción de software. Las interrupciones externas pueden desencadenarse por un borde que cae o sube o un nivel bajo. Esto es configurado por el Registro de Control de Interrupciones Externas A EICRA. Cuando las interrupciones externas están habilitadas y configuradas como activadas por nivel, las interrupciones se activan siempre que el pin se mantenga bajo.

El EICR del Registro de Control de Interrupciones Externas controla cómo funcionan las interrupciones externas.

Value	Description
00	The low level of INT0 generates an interrupt request.
01	Any logical change on INT0 generates an interrupt request.
10	The falling edge of INT0 generates an interrupt request.
11	The rising edge of INT0 generates an interrupt request.

Una interrupción de bajo nivel en el pin INT se detecta de forma asincrónica. Esto implica que esta interrupción se puede utilizar para despertar la parte también desde modos de suspensión distintos del modo inactivo. El reloj de E/S se detiene en todos los modos de suspensión, excepto en el modo inactivo.

Interrupción de cambio de pin

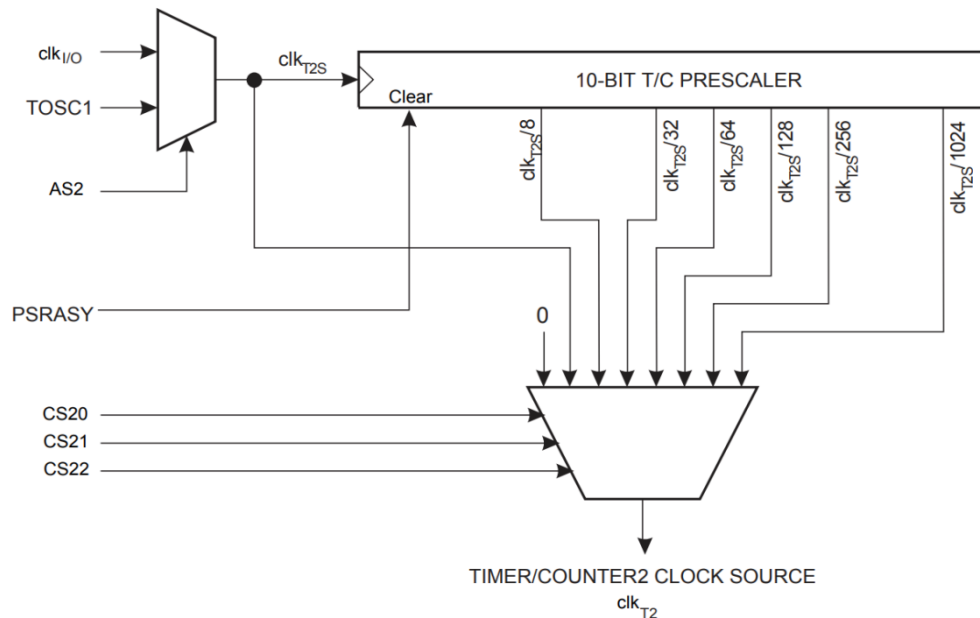
La solicitud de interrupción de cambio de pin se activa si cualquier pin PCINT habilitado cambia de estado. Hay varias interrupciones de cambio de pin, todas vinculadas a un conjunto de pines o puertos. En un dispositivo ATmega324PB, por ejemplo, las ubicaciones de cambio de pin son:

- La solicitud de interrupción de cambio de pin 4 (PCI4) se activa al cambiar los pines PCINT[38:32]
- La solicitud de interrupción de cambio de pin 3 (PCI3) se activa al cambiar los pines PCINT[31:24]
- La solicitud de interrupción de cambio de pin 2 (PCI2) se activa al cambiar los pines PCINT[23:16]
- La solicitud de interrupción de cambio de pin 1 (PCI1) se activa al cambiar los pines PCINT[15:8]
- La solicitud de interrupción de cambio de pin 0 (PCI0) se activa al cambiar los pines PCINT[7:0]

Los registros PCMSK4, PCMSK3, PCMSK2, PCMSK1y PCMSK0 controlan qué pines contribuyen a las interrupciones de cambio de pin. Las interrupciones de cambio de pin en PCINT se detectan de forma asincrónica. Esto implica que estas interrupciones también se pueden usar para despertar la parte de los modos de suspensión que no sean el modo inactivo.

Contador en tiempo real (RTC) de AVR

Los dispositivos AVR tienen un temporizador / contador tipo 2 (TC2) de uso general, doble canal, módulo de temporizador / contador de 8 bits. Este temporizador/contador permite la sincronización desde un cristal de reloj externo de 32 kHz, independiente de la E/S. Esto permite que el temporizador se ejecute como un contador de tiempo real (RTC) relativamente preciso.®



Fuente del reloj

La fuente de reloj para TC2 se denomina clk_{T2S} . Está conectado de forma predeterminada al reloj de E/S del sistema principal, $clk_{I/O}$. Escribiendo un 1 en el bit TC2 asíncrono en el Registro de estado asíncrono (ASSR. AS2), TC2 se sincroniza de forma asíncrona desde el pin TOSC1. Esto permite el uso de TC2 como RTC.

Cuando se establece AS2, los pines TOSC1 y TOSC2 se desconectan del puerto de E/S. Un cristal se puede conectar entre los pines TOSC1 y TOSC2 para servir como una fuente de reloj independiente para TC2. El oscilador está optimizado para su uso con un cristal de 32.768 kHz.

Preescalador

Para TC2, las posibles selecciones preescaladas son $clk_{T2S}/8$, $clk_{T2S}/32$, $clk_{T2S}/64$, $clk_{T2S}/128$, $clk_{T2S}/256$ y $clk_{T2S}/1024$. Además, se puede seleccionar clk_{T2S} , así como 0 (stop). El preescalador se restablece escribiendo un 1 en el bit TC2 de restablecimiento del escalador previo en el Registro de control general de TC2 (GTCCR. PSRASY). Esto permite al usuario operar con un preescalador definido.