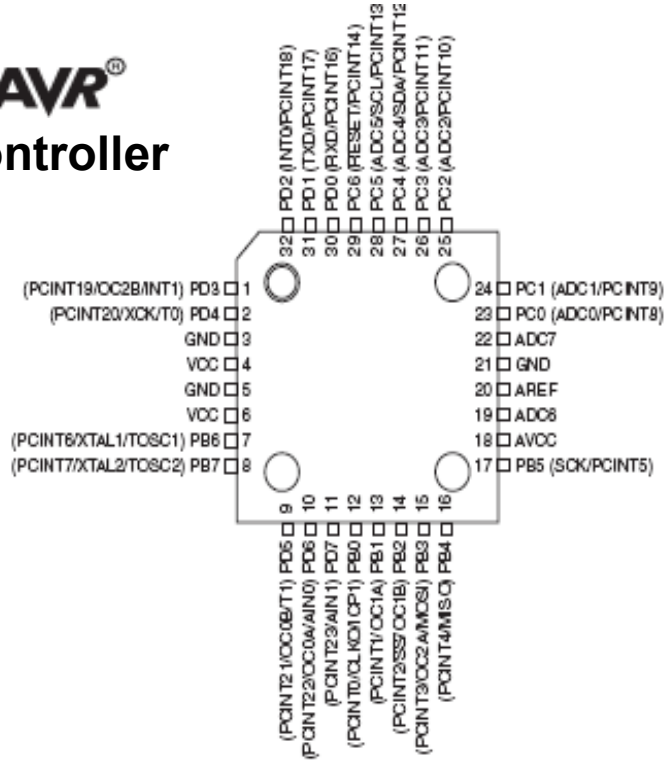
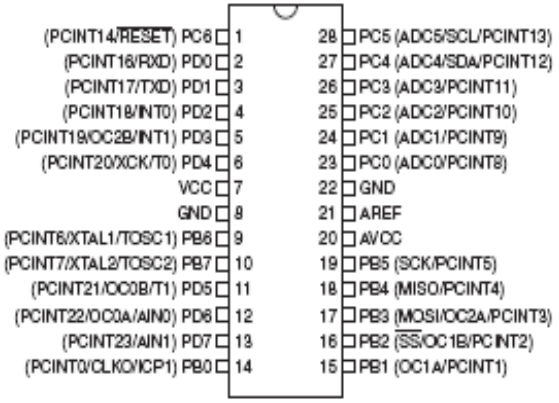
Tema 4: Ejemplo de un computador real: ATmegaX8pa

8-bit AVR®
Microcontroller





Índice

Introducción

- El ATmegaX8PA es un microcontrolador (MCU o µC) de la marca Atmel.
- Un microcontrolador es una pequeña computadora empotrada en un C.I. que contiene una CPU sencilla, unidad de reloj, memoria, y puertos de E/S (timers, IO ports, USARTs,..).
- Su reducido coste y tamaño lo hacen atractivo para un sin fin de aplicaciones: control de máquinas de automoción, dispositivos médicos, controles remotos, electrodomésticos, juguetes,...
- El AtmegaX8PA pertenece a la familia AVR8 que comparten una arquitectura común para todos los modelos pero con diferencias en cuanto a tipo de encapsulado, puertos y memoria.

Descripción general

- Arquitectura RISC. 8 bits
- Frecuencia de reloj de hasta 20 Mhz @ (4.5-5.5V)
- Hasta 20 Mips (20Mhz)
- Flash EEPROM X=4,8,16,32 Kb ATmegaX8pa
- Data SRAM 512/1K/2K bytes
- Data EEPROM 256/512 bytes

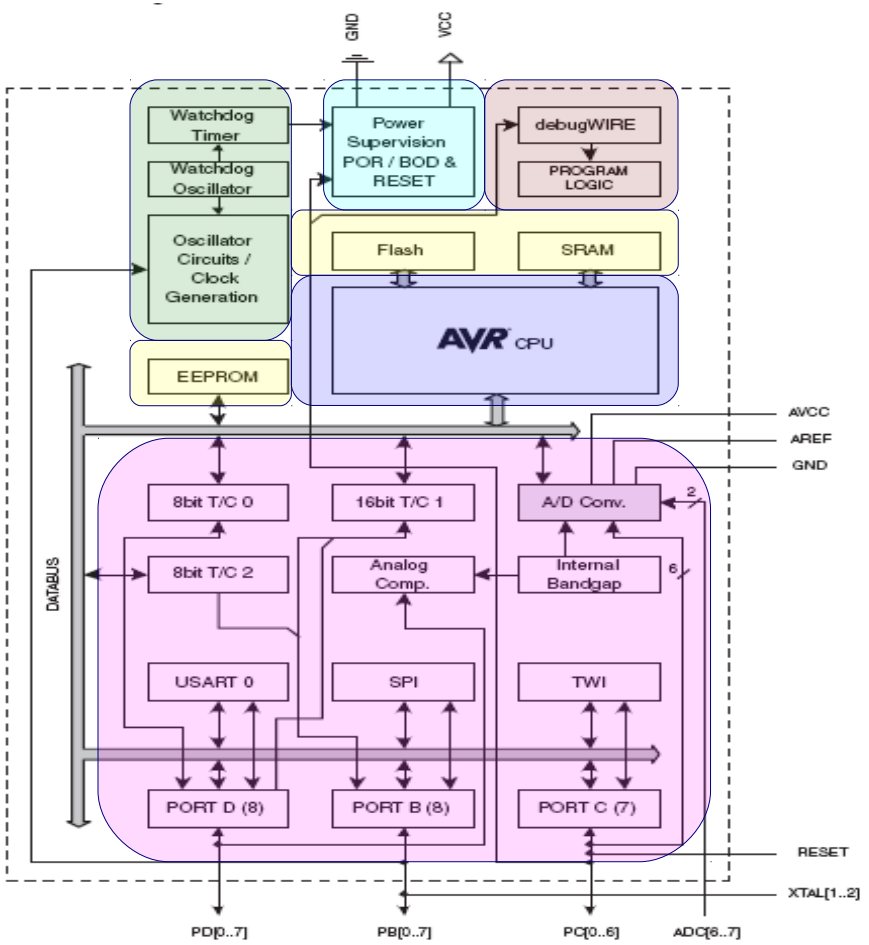
Descripción general

- Diversos periféricos
- Encapsulado PDIP, TQFP, MLF
- Versiones bajo consumo: 0-10 Mhz< @(2.7 -5.5V) 0-4 Mhz< @(1.8 -5.5V)
- Comparación entre procesadores

Procesador	*Flash*	*SRAM*	*Data Eeprom*	*Tamaño vector interrpción*
ATmega48pa	4Kb	256b	512b	1 instrucción
ATmega88pa	8Kb	512b	1Kb	1 instrucción
ATmega168pa	16Kb	512b	1Kb	2 instrucciones
ATmega328pa	32Kb	1Kb	2Kb	2 instrucciones

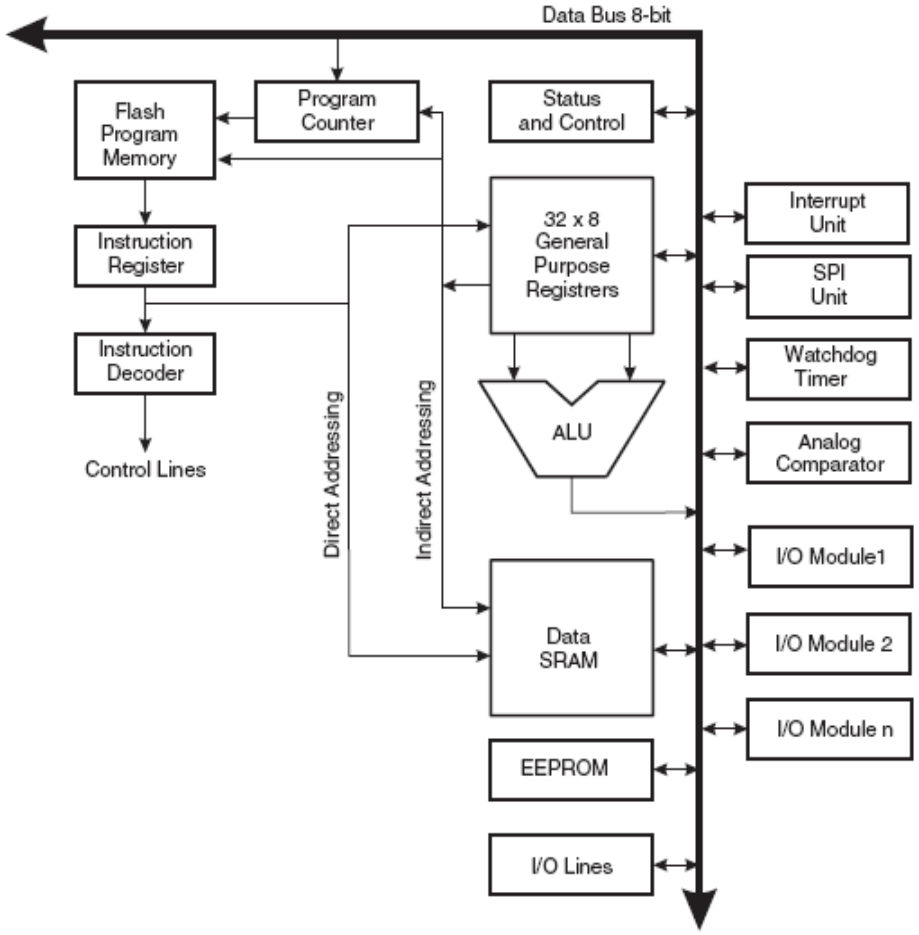
Arquitectura interna

- AVR-cpu
- Memorias (Flash, Sram,..)
- Generación de reloj
- Depuración
- Circuito de Reset
- Puertos de E/S:
  - CAD, Analog Comp.
  - Ports B,C,D
  - USART, TWI, SPI
  - Timers



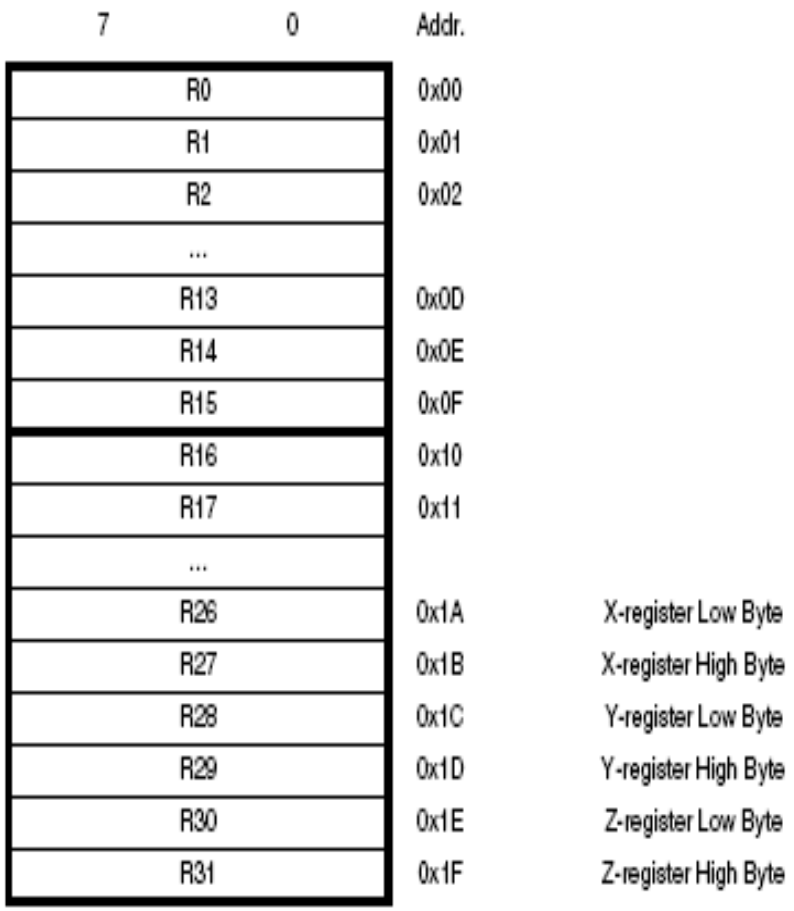
Arquitectura interna

- Arquitectura Harvard
- Pipeline de 1 nivel:
  - Execute y pre-fetch simultáneos
- Bus de datos de 8 bits
- Banco de registros
- Arquitectura RISC
  - Load/Store
- ALU con multiplicador



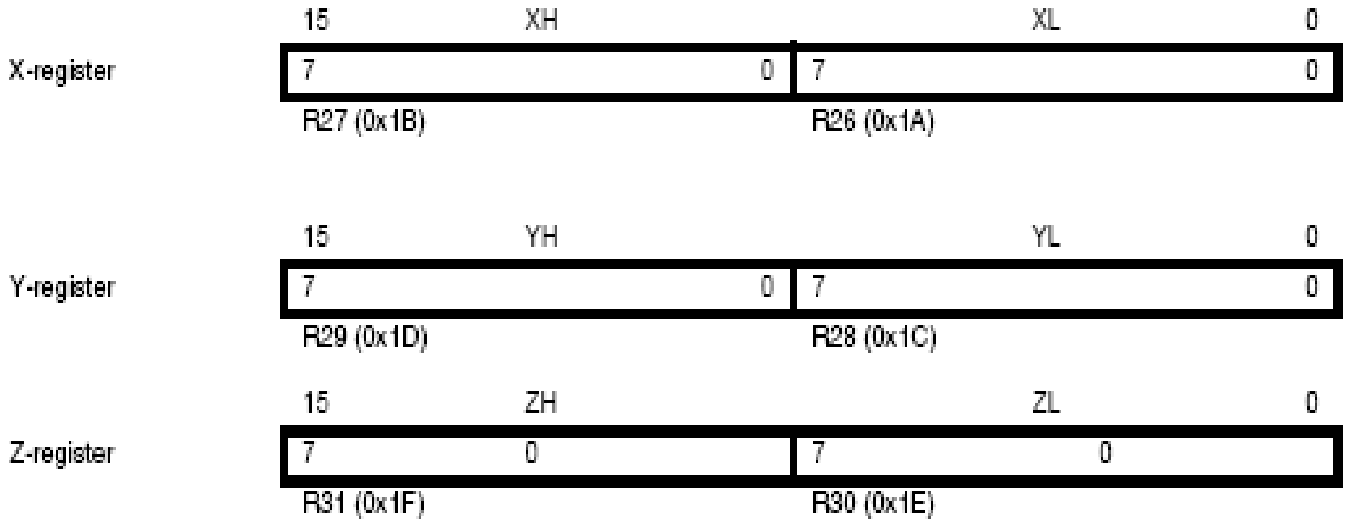
Arquitectura interna

- Registros de propósito general
  - Ocupan las 32 primeras posiciones de la memoria.
  - Instrucciones con datos de tipo inmediato sólo pueden usar los 16 superiores.
  - Registros X, Y y Z de 16 bits para los modos de direccionamiento indirectos.



Arquitectura interna

- Registros X, Y, Z
  - Se comportan como registros de 16 bits cuando se usan.
  - Utilizados para los modos de direccionamiento indirectos.



Arquitectura interna

- Registro de estado
  - Se encuentra en el área de entrada/salida de la memoria de datos.
  - Contiene los banderines (flags) que reflejan el resultado de la ejecución de algunas instrucciones (principalmente aritméticas, lógicas, etc.)
    - Bit C (Acarreo)
    - Bit Z (Zero) . Se pone a 1 para resultado cero, 0 en otro caso.
    - Bit N (Negative). Bit 7 del resultado.

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Arquitectura interna

- Registro de estado (cont.)
  - Bit V (Overflow en Ca2) Si A,B son operandos y R resultado, este bit se calcula como:
    - $V = A7\ B7\ \overline{R7} + \overline{A7}\ \overline{B7}\ R7$ en suma
    - $V = A7\ \overline{B7}\ \overline{R7} + \overline{A7}\ B7\ R7$ en resta
  - Bit S (Signo) Refleja el signo correcto del resultado en operaciones en Ca2.
    - $S = N \oplus V$

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Arquitectura interna

- Registro de estado (cont.)
  - Bit H (Half Carry) Bit de acarreo de la etapa 3 de la ALU (semiacarreo).
  - Bit T. Bit de propósito general.
  - Bit I (Interrupción). Permite la generación de interrupciones cuando es 1 y las enmascara cuando es 0.

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Arquitectura interna

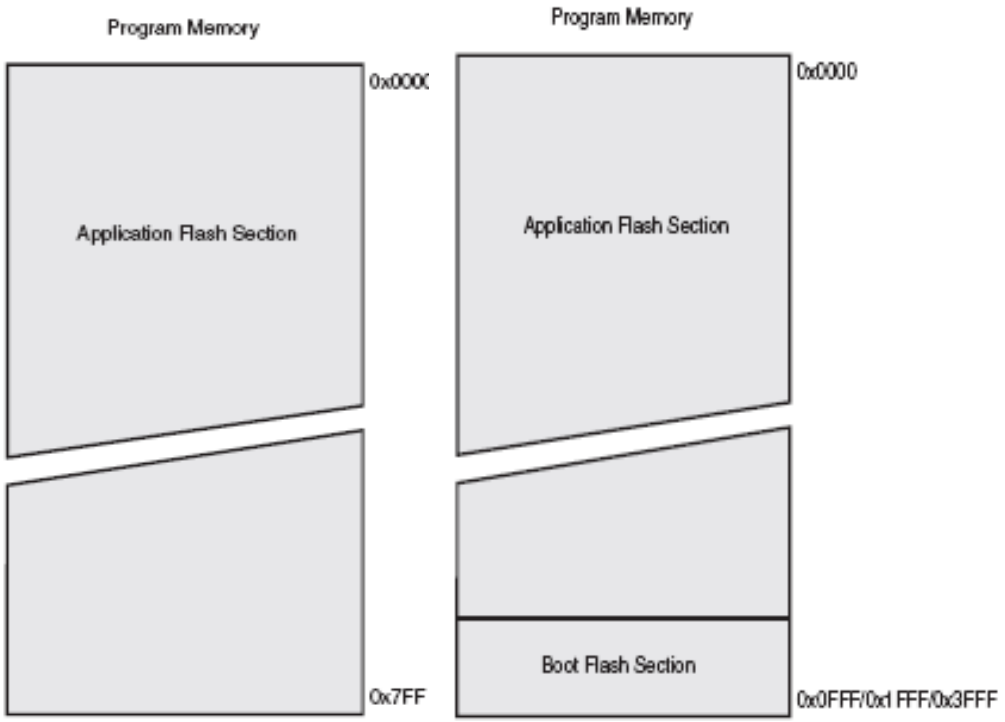
- Puntero de Pila (SPH y SPL)
  - Se encuentra en la región de memoria asignada a E/S.
  - Apunta al área de pila (Valor inicial RAMEND).
  - El SP se decrementa una unidad al meter un dato en pila (PUSH) pila y se incrementa una unidad al sacarlo (POP). El SP apunta a la dirección siguiente a la cima de la PILA.
  - Las llamadas a subrutinas o rutinas de interrupción (RCALL, etc) hacen que el SP se decremente en dos unidades y para los retornos (RET, RETI), el puntero se incrementa en dos unidades.

Bit	15	14	13	12	11	10	9	8	
0x3E (0x5E)	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
0x3D (0x5D)	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	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	
	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	

Organización de la memoria

- Memoria de programa (Flash)
  - Atmega48pa
  - Resto dos secciones:
    - Arranque “Boot”
    - Aplicación
  - Cada renglón es

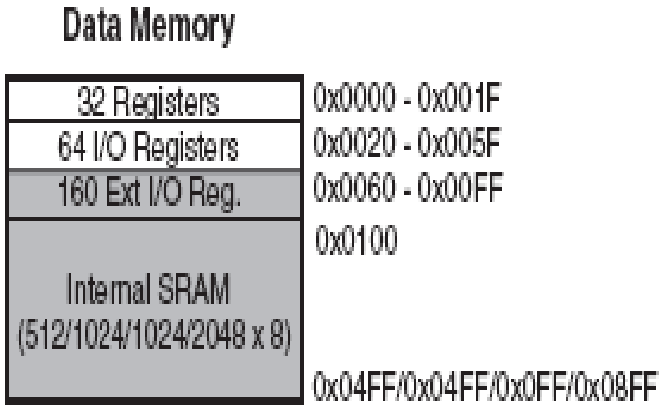
de tamaño Word



Organización de la memoria

- Memoria de datos
  - 32 registros de propósito general (todos los modos de direccionamiento)
  - Registros de E/S
    - 64 I/O Registers (instrucciones IN, OUT, modo directo e indirecto)
    - 160 Ext Registers

    (modos directo e indirecto)
  - SRAM (modos directo e indirecto)

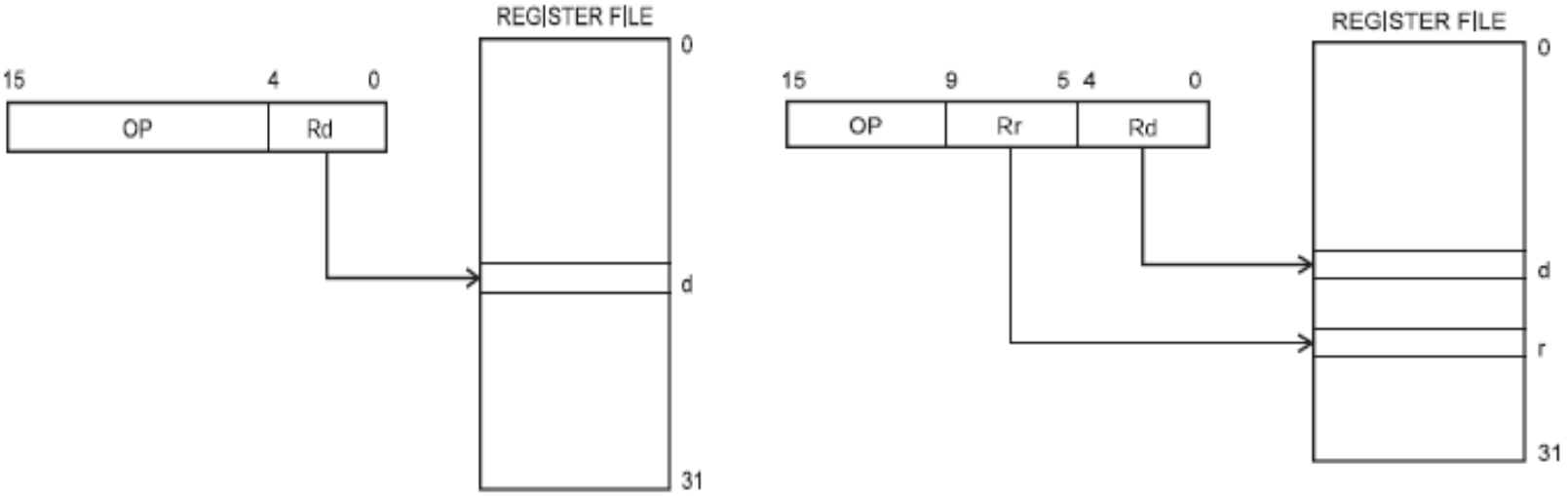


Modos de direccionamiento

- Modos de direccionamiento para memoria de datos
  - Registro directo
  - Directo
  - Indirectos
    - Indirecto
    - Indirecto con predecremento
    - Indirecto con postincremento
    - Indirecto con desplazamiento
  - Inmediato

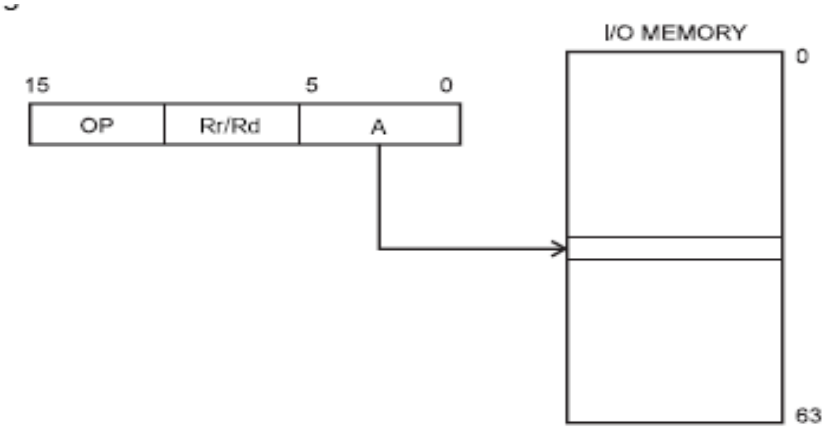
Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont.)
  - Directo: la instrucción define el registro o registros cuyo contenido se verá afectado por la propia instrucción.
    - Ejemplos: COM R4 MOV R1,R2



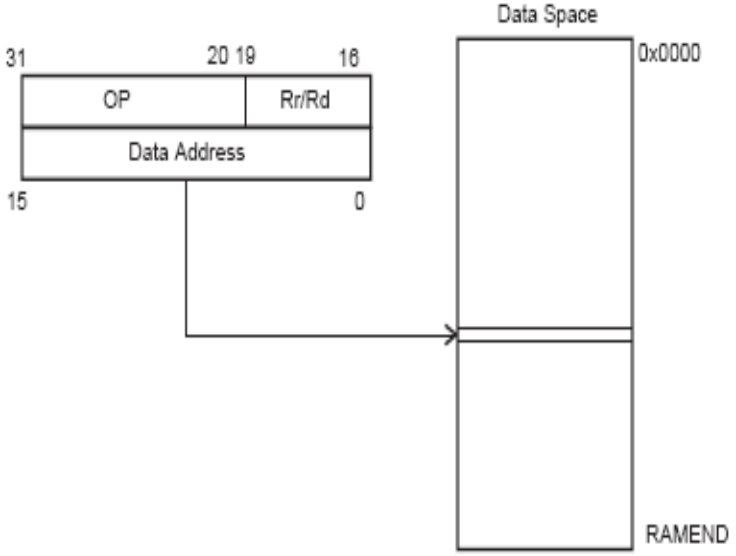
Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont)
  - Registro E/S: la instrucción define el registro de E/S y el registro del banco afectados (Sólo instrucciones IN y OUT).
    - Ejemplos: IN R1, 56 ; OUT 21, R10



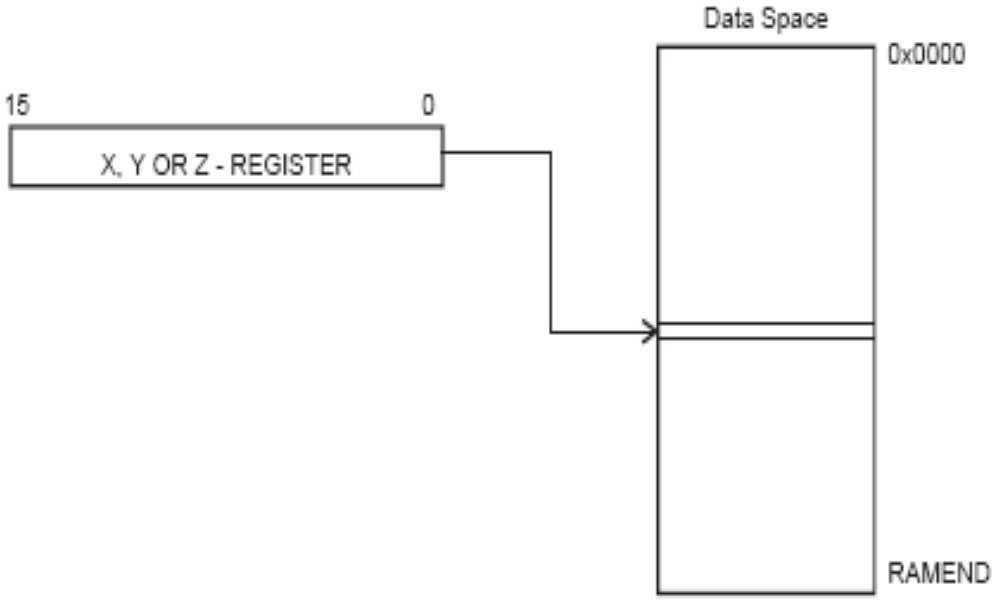
Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont)
  - Directo: La instrucción contiene la dirección de memoria (16bits) del dato. Además contiene un campo (Rd/Rr) que identifica el registro destino o fuente. Instrucciones LDS, STS.
    - Ejemplos:
      - LDS R23,$D0
      - STS $12,R1



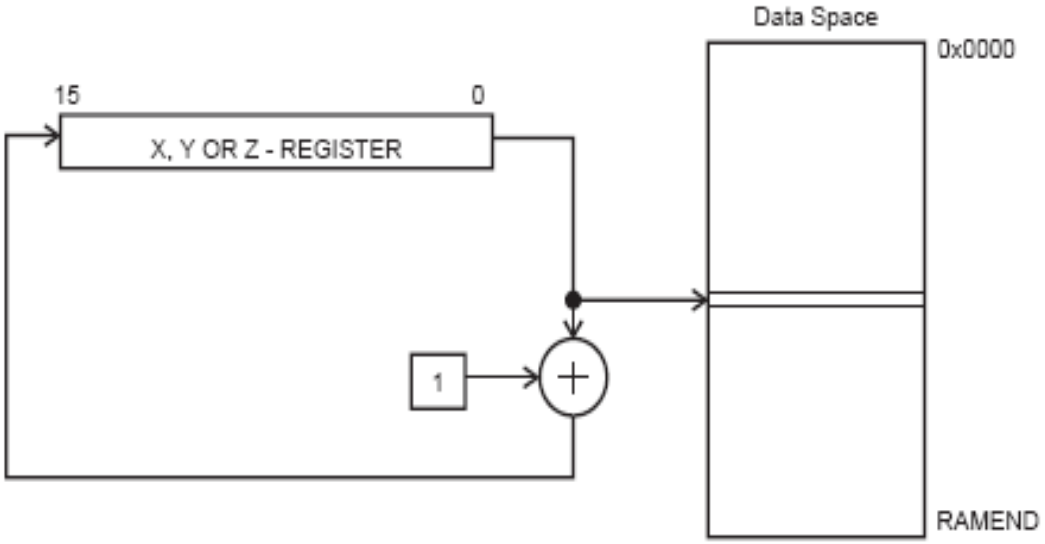
Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont)
  - Indirecto: La instrucción referencia al registro X,Y o Z que contiene la dirección del operando.
    - Ejemplos:
      - LD R1,X
      - ST Z,R10



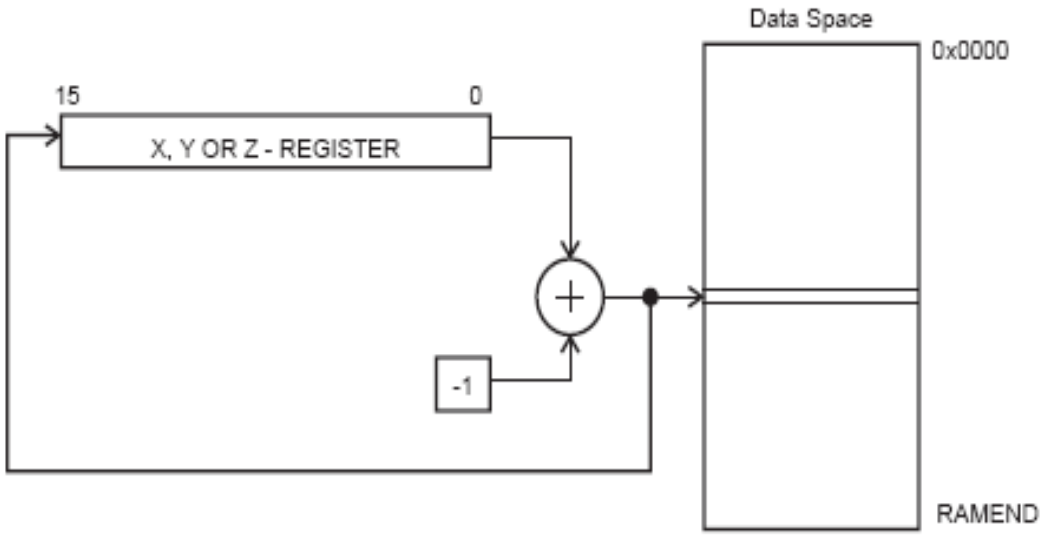
Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont)
  - Indirecto con postincremento: La instrucción referencia al registro X,Y o Z que contiene la dirección del operando, que después se incrementa en una unidad.
  - Ejemplos:
    - LD R0,X+
    - ST Z+,R1



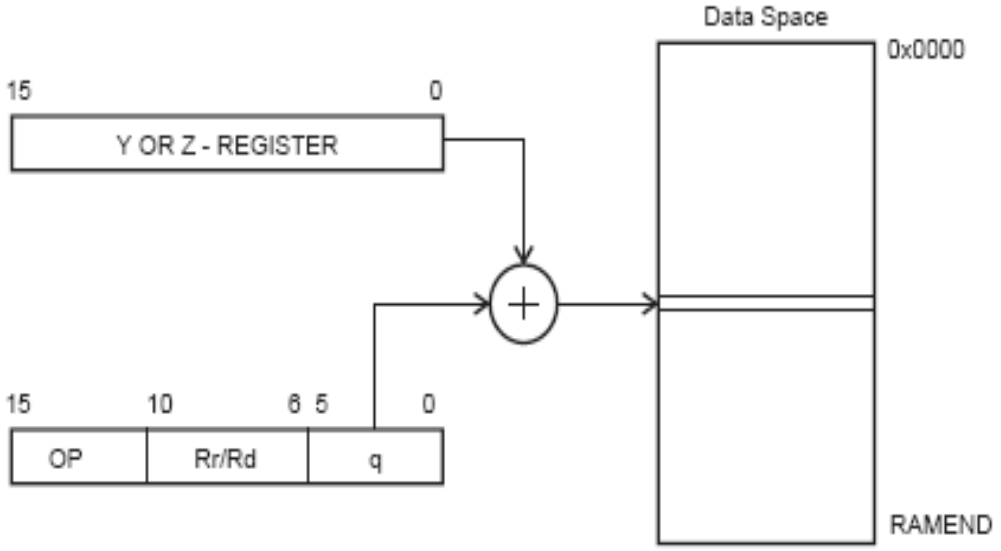
Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont)
  - Indirecto con predecremento: La instrucción referencia al registro X,Y o Z que tras decrementarse en una unidad, contiene la dirección del operando.
  - Ejemplos:
    - LD R0,-X
    - ST -Z,R1



Modos de direccionamiento

- Modos de direccionamiento para memoria de datos (cont)
  - Indirecto con desplazamiento: La dirección del dato se obtiene sumando el desplazamiento q (0<q<64) a la dirección contenida en el registro Y o Z. El resultado no se actualiza en el registro.
  - Ejemplos:
    - LDD R0,Y+10
    - STD Z+9,R1



Modos de direccionamiento

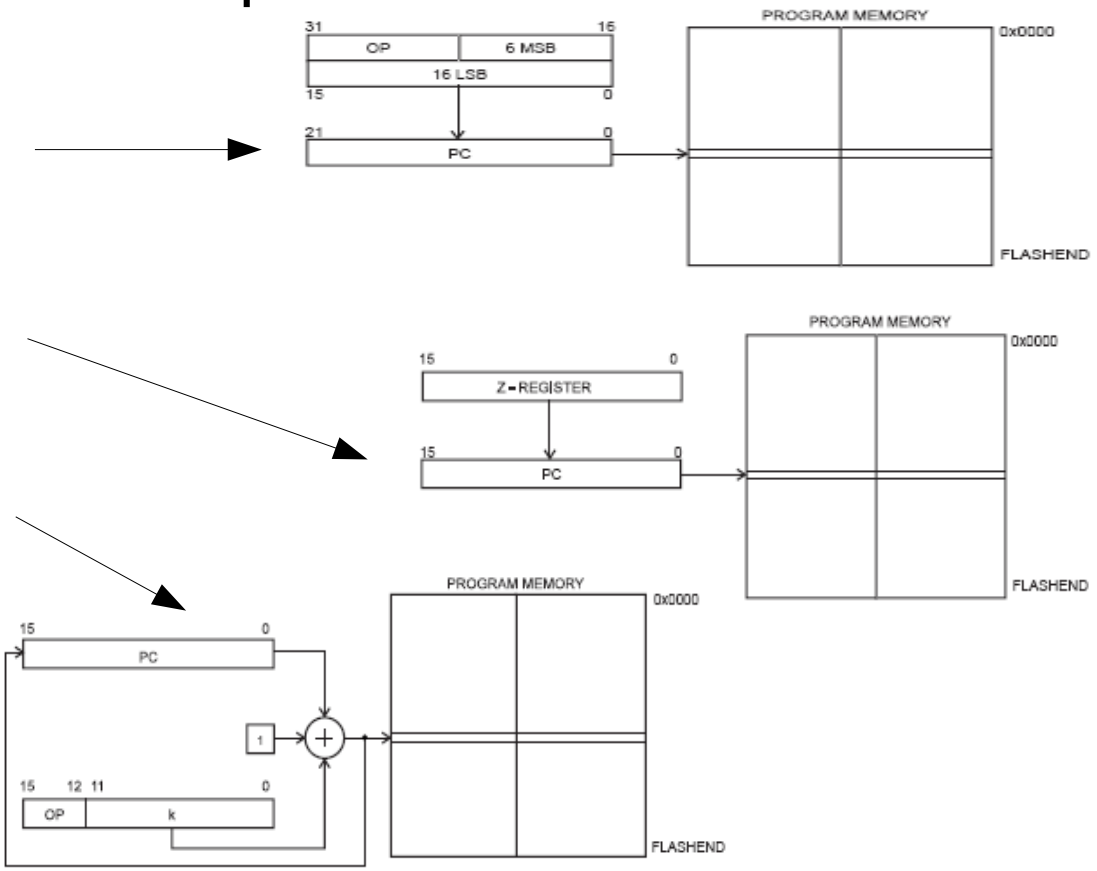
- Modos de direccionamiento para memoria de datos (cont)
  - Inmediato: El dato está está codificado en la propia instrucción
    - Ejemplos: LDI r16,255, ANDI r25,0x10

Modos de direccionamiento

- Modos de direccionamiento para memoria de programa
  - Constantes de programa (fuera del alcance de esta asignatura).
    - Indirecto
    - Indirecto con postincremento
  - Instrucciones
    - Directo
    - Indirecto
    - Relativo

Modos de direccionamiento

- Modos de direccionamiento para memoria de programa
  - Instrucciones Directo.
    - JMP y CALL
  - Instrucciones Indirecto
    - IJMP e ICALL
  - Instrucciones Relativo
    - RJMP y RCALL
    - [-2048 , 2047]



Juego de instrucciones

- Instrucciones de transferencia de datos
- Instrucciones aritmético-lógicas
- Instrucciones de salto
- Instrucciones de manejo de bits
- Instrucciones de control del sistema

Juego de instrucciones

- Instrucciones sin operandos
  - Mnemónico
- Instrucciones con un operando
  - Mnemónico operando
- Instrucciones con dos operandos
  - Mnemónico op-destino,op-fuente
- Para cada instrucción se presentará la siguiente información

Mnemónico	Operandos	Descripción	Rango	Operación	Banderines	Ciclos de reloj

Juego de instrucciones

- Instrucciones de transferencia de datos

MOV	Rd,Rr	Copiar registro	d,r∈[0,31]	Rd←Rr	Ninguno	1
MOVW	Rd,Rr	Copiar registro W	d,r∈[0,30]	Rd+1:Rd←Rr+1:Rr	Ninguno	1
LDI	Rd,k	Cargar dato inmediato	d∈[16,31] k∈[0,255]	Rd← k	Ninguno	1
LDS	Rd,k	Cargar dato desde la memoria	d∈[0,31] k<64K	Rd← (k)	Ninguno	2
LD	Rd,X Rd,X+ Rd,-X Rd,Y Rd,Y+ Rd,-Y Rd,Z Rd,Z+ Rd,-Z	Carga el registro con un dato indirecto	d∈[0,31]	Rd← (X) Rd← (X);X←X+1 X←X-1-,Rd← (X) Rd← (Y) Rd← (Y);Y←Y+1 Y←Y-1;Rd← (Y) Rd← (Z) Rd← (Z);Z←Z+1 Z←Z-1;Rd← (Z)	Ninguno	2
LDD	Rd,Y+q Rd,Z+q	Carga el registro con un dato indirecto con desplazamiento	d∈[0,31] q∈[0,63]	Rd← (Y+q) Rd← (Z+q)	Ninguno	2

Juego de instrucciones

- Instrucciones de transferencia de datos

STS	K, Rr	Almacenar dato en memoria	r∈ [0,31] K∈ [0,64k]	(K)← Rr	Ninguno	2
ST	X,Rr X+,Rr -X,Rr Y,Rr Y+,Rr -Y,Rr Z,Rr Z+,Rr -Z,Rr	Almacenar registro en memoria	r∈ [0,31]	(X)← Rr (X)← Rr;X←X+1 X←X-1-, (X)←Rr (Y)←Rr (Y)← Rr;Y←Y+1 Y←Y-1; (Y)← Rr (Z)← Rr (Z)←Rr;Z←Z+1 Z←Z-1; (Z)←Rr	Ninguno	2
STD	Y+q,Rr Z+q,Rr	Almacenar registro en memoria con indirecto con desplazamiento	r∈ [0,31]	(Y+q)←Rr (Z+q)← Rr	Ninguno	2
LPM	Rd,Z Rd,Z+	Carga memoria de programa		R0← (Z) Rd← (Z) Rd← (Z); Z←Z+1	Ninguno	3
SPM		Almacenar en memoria de programa		(Z) ← R1:R0	Ninguno	-

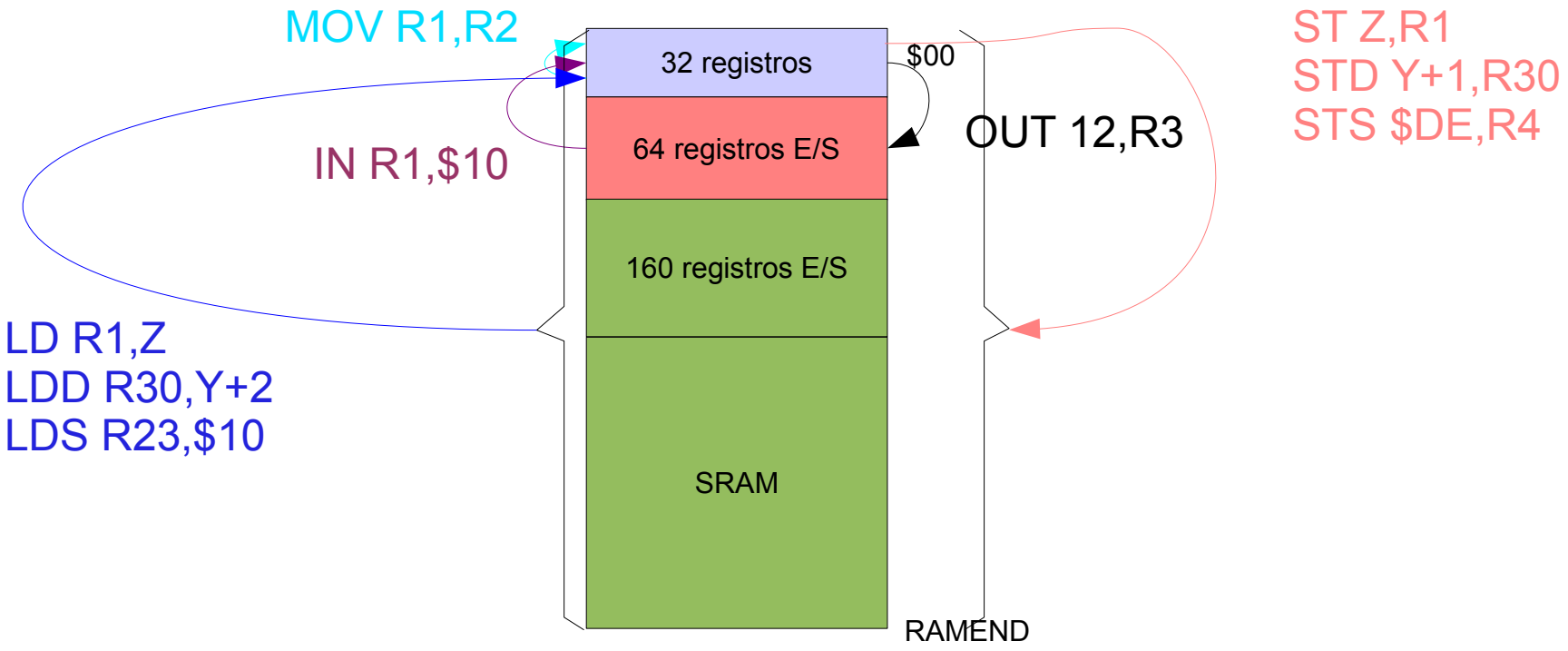
Juego de instrucciones

- Instrucciones de transferencia de datos

IN	Rd,P	Entrada del puerto	d∈ [0,31] P∈ [0,63]	Rd←P	Ninguno	1
OUT	P,Rr	Salida hacia el puerto	r∈ [0,31] P∈ [0,63]	P← Rr	Ninguno	1
PUSH	Rr	Empujar en pila	r∈ [0,31]	STACK←Rr	Ninguno	2
POP	Rd	Sacar de pila	d∈ [0,31]	Rd← STACK	Ninguno	2

Juego de instrucciones

- Instrucciones de transferencia de datos



Juego de instrucciones

- Instrucciones aritmético-lógicas

ADD	Rd,Rr	Suma sin carry	d,r∈[0,31]	Rd← Rd+Rr	Z,N,V,C,H	1
ADC	Rd,Rr	Suma con carry	d,r∈[0,31]	Rd← Rd+Rr+C	Z,N,V,C,H	1
ADIW	Rd,K	Suma inmediato con palabra	d∈[24,26,28,30] K∈[0,63]	Rd+1:Rd← Rd+1:Rd + K	Z,N,V,C	2
SUB	Rd,Rr	Resta sin carry	d,r∈[0,31]	Rd←Rd-Rr	Z,N,V,C,H	1
SUBI	Rd,K	Resta inmediato	d∈[16,31] K∈[0,255]	Rd←Rd-K	Z,N,V,C,H	1
SBC	Rd,Rr	Resta con carry	d,r∈[0,31]	Rd←Rd-Rr-C	Z,N,V,C,H	1
SBCI	Rd,K	Resta inmediato con carry	d∈[16,31] K∈[0,255]	Rd←Rd-K-C	Z,N,V,C,H	1
SBIW	Rd,K	Resta inmediato con palabra	d∈[24,26,28,30] K∈[0,63]	Rd+1:Rd← Rd+1:Rd - K	Z,N,V,C	2
AND	Rd,Rr	And lógica	d,r∈[0,31]	Rd←Rd∧Rr	Z,N,V	1
ANDI	Rd,K	And lógica con dato inmediato	d∈[16,31] K∈[0,255]	Rd←Rd∧K	Z,N,V	1
OR	Rd,Rr	Or lógica	d,r∈[0,31]	Rd←Rd∨Rr	Z,N,V	1
ORI	Rd,K	Or lógica con dato inmediato	d∈[16,31] K∈[0,255]	Rd←Rd∨K	Z,N,V	1
EOR	Rd,Rr	Exclusive or	d,r∈[0,31]	Rd←Rd⊕Rr	Z,N,V	1
COM	Rd	Complemento a 1	d,r∈[0,31]	Rd←$FF-Rd	Z,N,V,C	1
NEG	Rd	Complemento a 2	d,r∈[0,31]	Rd←$00-Rd	Z,N,V,C	1
INC	Rd	Incrementa	d,r∈[0,31]	Rd←Rd+1	Z,N,V	1
DEC	Rd	Decrementa	d,r∈[0,31]	Rd←Rd-1	Z,N,V	1

Juego de instrucciones

- Instrucciones aritmético-lógicas

CLR	Rd	Poner a cero	d,r∈ [0,31]	Rd←0	Z,N,V	1
SER	Rd	Poner todo a 1	d,r∈ [0,31]	Rd←$FF	Z,N,V	1
CP	Rd,Rr	Compara	d,r∈ [0,31]	Rd-Rr	Z,N,V,C,H	1
CPC	Rd,Rr	Compara con carry	d,r∈ [0,31]	Rd-Rr-C	Z,N,V,C,H	1
CPI	Rd,K	Compara inmediato	d∈ [16,31] K∈ [0,255]	Rd-K	Z,N,V,C,H	1
MUL	Rd,Rr	Multiplica sin signo	d,r∈ [0,31]	R1:R0←Rd x Rr	Z,C	2
MULS	Rd,Rr	Multiplica con signo	d,r∈ [0,31]	R1:R0←Rd x Rr	Z,C	2
MULSU	Rd,Rr	Multiplica signo con sin signo	d,r∈ [0,31]	R1:R0←Rd x Rr (Rd signed Rr unsigned)	Z,C	2

Juego de instrucciones

- Instrucciones de salto.

RJMP	Etiqueta	Salto relativo	-2k< Etiqueta <2k	PC←PC+ Etiqueta +1	Ninguno	2
JMP (1)	Etiqueta	Salto	0<Etiqueta<4M	PC← Etiqueta	Ninguno	2
IJMP		Salto indirecto		PC ← (Z)	Ninguno	3
RCALL	Etiqueta	Llamada a subrutina relativa	-2k< Etiqueta <2k	STACK ← PC PC←PC+ Etiqueta +1	Ninguno	3
CALL (1)	Etiqueta	Llamada a subrutina	0<Etiqueta<4M	STACK ← PC PC← Etiqueta	Ninguno	3
ICALL		Llamada a subrutina indirecta		STACK ← PC PC ← (Z)	Ninguno	4
RET		Regreso de subrutina		PC← STACK	Ninguno	4
RETI		Regreso de interrup.		PC← STACK	I	4
CPSE	Rd,Rr	Compara, esquiva si iguales	d,r∈ [0,31]	Si Rd=Rr PC← PC+2 (ó 3)	Ninguno	01/02/03
SBRC	Rr,b	Esquiva si el bit está a cero	r∈ [0,31] b∈ [0,7]	Si (Rd(b)=0) PC← PC+2 (ó 3)	Ninguno	01/02/03
SBRS	Rr,b	Esquiva si el bit está a uno	r∈ [0,31] b∈ [0,7]	Si (Rd(b)=1) PC← PC+2 (ó 3)	Ninguno	01/02/03

(1) Sólo disponibles para ATMEGA168PA y ATMEGA328PA

Juego de instrucciones

- Instrucciones de salto.

SBIC	P,b	Esquiva si el bit del puerto está a 0	P∈ [0,31] b∈ [0,7]	Si (I/o(P,b)=0) PC← PC+2 (ó 3)	Ninguno	1 o 2 o 3
SBIS	P,b	Esquiva si el bit del puerto está a1	P∈ [0,31] b∈ [0,7]	Si (I/O(P,b)=1) PC← PC+2 (ó 3)	Ninguno	1 o 2 o 3
BREQ	Etiqueta	Salta si iguales	Etiqueta ∈ [-64,63]	Si (Z=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRNE	Etiqueta	Salta si distintos	Etiqueta ∈ [-64,63]	Si (Z=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRCS	Etiqueta	Salta si C está a 1	Etiqueta ∈ [-64,63]	Si (C=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRCC	Etiqueta	Salta si Cestá a 0	Etiqueta ∈ [-64,63]	Si (C=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRSH	Etiqueta	Salta si igual o mayor	Etiqueta ∈ [-64,63]	Si (C=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2

Juego de instrucciones

- Instrucciones de salto.

BRLO	Etiqueta	Salta si menor	Etiqueta ∈ [-64,63]	Si (C=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRMI	Etiqueta	Salta si negativo	Etiqueta ∈ [-64,63]	Si (N=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRPL	Etiqueta	Salta si positivo	Etiqueta ∈ [-64,63]	Si (N=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRHS	Etiqueta	Salta si H está a 1	Etiqueta ∈ [-64,63]	Si (H=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRHC	Etiqueta	Salta si H está a 0	Etiqueta ∈ [-64,63]	Si (H=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRTS	Etiqueta	Salta si T está a 1	Etiqueta ∈ [-64,63]	Si (T=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRTC	Etiqueta	Salta si T está a 0	Etiqueta ∈ [-64,63]	Si (T=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRVS	Etiqueta	Salta si V está a 1	Etiqueta ∈ [-64,63]	Si (V=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRVC	Etiqueta	Salta si V está a 0	Etiqueta ∈ [-64,63]	Si (V=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRIE	Etiqueta	Salta si I está a 1	Etiqueta ∈ [-64,63]	Si (I=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRID	Etiqueta	Salta si I está a 0	Etiqueta ∈ [-64,63]	Si (I=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2

Juego de instrucciones

- Instrucciones de salto.

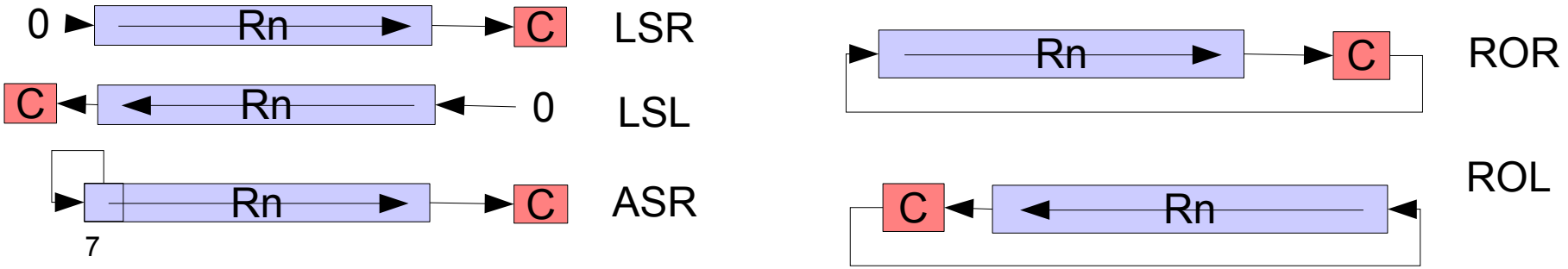
BRGE	Etiqueta	Salta si mayor o igual, (signo)	Etiqueta ∈ [-64,63]	Si (N⊕V=0) PC← PC+ Etiqueta +1	Ninguno	1 o 2
BRLT	Etiqueta	Salta si menor (signo)	Etiqueta ∈ [-64,63]	Si (N⊕V=1) PC← PC+ Etiqueta +1	Ninguno	1 o 2

Test (CP Rd,Rr)	Booleana	Mnemonico	Comentario
Rd ≥Rr	(N ⊕ V) = 0	BRGE	Signo
Rd < Rr	(N ⊕ V) = 1	BRLT	Signo
Rd = Rr	Z = 1	BREQ	Signo/Sin signo
Rd ≠ Rr	Z = 0	BRNE	Signo/Sin signo
Rd ≥ Rr	C = 0	BRCC/BRSH	Sin signo
Rd < Rr	C = 1	BRCS/BRLO	Sin signo
Carry	C=1	BRCS	Simple
Sin carry	C=0	BRCC	Simple
Negativo	N=1	BRMI	Simple
Positivo	N=0	BRPL	Simple
Overflow	V=1	BRVS	Simple
Sin overflow	V=0	BRVC	Simple
Cero	Z=1	BREQ	Simple
No cero	Z=0	BRNE	Simple

Juego de instrucciones

- Instrucciones de bit y de bit-test.

LSL	Rd	Desplazamiento a la izquierda	d∈ [0,31]	Rd(n+1)←Rd(n), Rd(0)←0, C← Rd(7)	Z,C,N,V,H	1
LSR	Rd	Desplazamiento a la derecha	d∈ [0,31]	Rd(n)←Rd(n+1), Rd(7)←0, C← Rd(0)	Z,C,N,V	1
ROL	Rd	Rotación a la izquierda	d∈ [0,31]	Rd(n+1)←Rd(n), Rd(0)←C, C← Rd(7)	Z,C,N,V,H	1
ROR	Rd	Rotación a la derecha	d∈ [0,31]	Rd(n)←Rd(n+1), Rd(7)←C, C← Rd(0)	Z,C,N,V,	1
ASR	Rd	Desplazamiento aritmético a la derecha	d∈ [0,31]	Rd(n)←Rd(n+1), Rd(7)←Rd(7), C← Rd(0)	Z,C,N,V,	1



Juego de instrucciones

- Instrucciones de bit y de bit-test.

SWAP	Rd	Intercambia nibbles	d∈ [0,31]	Rd(3..0)←→Rd(7.4)	Ninguno	1
SBI	P,b	Poner a 1 el bit b del puerto IO	b∈ [0,7] P∈ [0,31]	IO(P,b)←1	Ninguno	2
CBI	P,b	Poner a 0 el bit b del puerto IO	b∈ [0,7] P∈ [0,31]	IO(P,b)←0	Ninguno	2
SEcc	[1]	Poner a 1 el bit cc del registro de estado			cc	1
CLcc	[1]	Poner a o el bit cc del registro de estado			cc	1

cc= C,N,T,Z,I,V,H,S

Juego de instrucciones

- Instrucciones de control

NOP		Nada			Ninguno	1
BREAK		Para depuración			Ninguno	N/A
WDR		Reinica el temporizador del perro guardián			Ninguno	1
SLEEP		Dormir			Ninguno	1

Directivas de ensamblador

- Directivas de ensamblador

  Son comandos al programa que genera el código objeto y que se encuentran mezclados en el fichero fuente con las instrucciones del microcontrolador .

  - CSEG-Code Segment
    - Sintaxis: .CSEG
  - DSEG-Date Segment
    - Sintaxis: .DSEG

Directivas de ensamblador

- Directivas de ensamblador(cont.)
  - BYTE – Reserva bytes a una variable
    - Reserva en memoria de datos. Posible sólo en DSEG
    - Sintaxis: label: .BYTE expresion
    - *Var1: .BYTE 1*
    - *Tabla: .BYTE 10*

Directivas de ensamblador

- Directivas de ensamblador(cont.)
  - DEF – Asigna un nombre simbólico a un registro.
    - Sintaxis: .DEF symbol=register
    - *.DEF temp = r16*
    - *.DEF ior= r0*
  - EQU – Símbolo igual a expresión
    - Sintaxis: .EQU label = expression
    - *.EQU puertas = 2*

Directivas de ensamblador

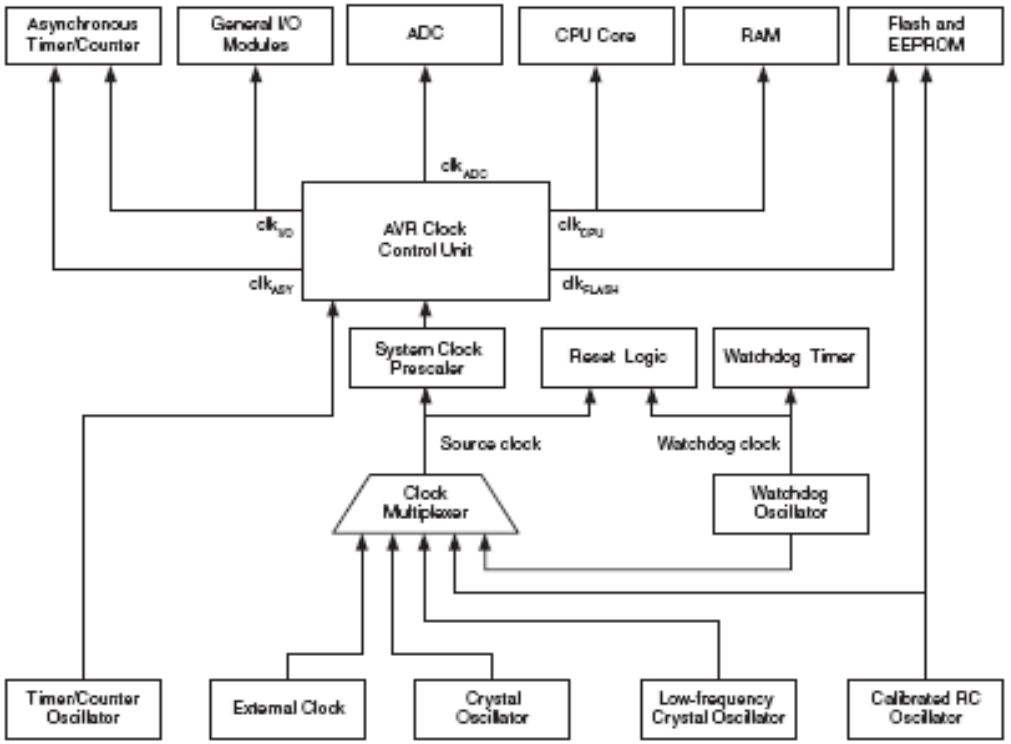
- Directivas de ensamblador(cont.)
  - ORG – Establece la dirección de memoria

    Sintaxis: .ORG expression

```
.DSEG
.ORG 0X37      ;Dirección $37 de la memoria de datos
               ;Si no se indica dirección, por defecto $60
Variable: .BYTE 1
.CSEG
.ORG 0x10
Mov  r0,r1
```

Arquitectura interna

- Reloj del sistema y opciones de reloj
  - Fuentes
    - Externo
    - Oscilador de cristal
    - Oscilador RC calibrado
    - Oscilador del perro guardián.
  - Actúan sobre:
    - AVR clock control unit mediante un Prescaler
    - Circuito de Reset



Arquitectura interna

- Reloj del sistema y opciones de reloj (cont.)
  - La fuente de reloj se configura mediante la programación de unos “fusibles”.
  - De fábrica RC a 8MHz con Prescaler de 8.
  - Secuencia de puesta en marcha del reloj (Clock startup sequence)
    - Esperar un voltaje adecuado de Vcc.
      - Circuito de RESET y un posterior Timeout llevado a cabo por el Watchdog Timer.
    - Un número previo de oscilaciones.
      - Fusibles

Arquitectura interna

- Reloj del sistema y opciones de reloj(cont.)
  - Unidad de control y distribución de reloj
    - Permite apagar aquellos módulos que no sean usados.
    - Adapta el consumo al tipo de aplicación.
    - Modos bajo consumo (instrucción Sleep)

Arquitectura interna

- Circuito de Reset
  - Provoca que el sistema pase a un estado inicial conocido.
  - Los registros de ES toman sus valores por defecto.
  - Se busca la instrucción en la posición asociada al vector de RESET.
  - Existen varias causas que generan un RESET:
    - Power on Reset
    - Reset externo
    - Perro guardián (Watchdog Reset)
    - Detector de “apagones” (Brown-out detectors)

Interrupciones

- ¿Qué es una interrupción?
  - Evento que requiere la suspensión (interrupción) del programa actual y la ejecución de una rutina concreta (rutina de interrupción), al final de la cual se devuelve el control al programa interrumpido.
- ¿Qué se necesita para procesar interrupciones?
  - Pila. Almacena la dirección de la instrucción del programa interrumpido.
  - Rutina de interrupción ***instalada*** debidamente según su vector de interrupción.
  - Guardar la información de estado de la CPU.

Interrupciones

- El SP del AtmegaX8pa se inicia automáticamente a la posición más alta de la memoria de datos.
- La instalación de la rutina de interrupción requiere situar la instrucción jmp o rjmp en la posición adecuada del vector de interrupción.
  - Modelos Atmega168pa y Atmega328pa requiere instrucción JMP.
  - Modelos Atmega48pa y Atmega88pa requiere instrucción RJMP.
- La tabla de vectores de interrupción depende del modelo.

Interrupciones (Tabla vectores ATmega168pa)

VectorNo.	Program Address(2)	Source	Interrupt 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	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Counter2 Overflow
11	0x0014	TIMER1 CAPT	Timer/Counter1 Capture Event
12	0x0016	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1 COMPB	Timer/Coutner1 Compare Match B
14	0x001A	TIMER1 OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x0020	TIMER0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete
19	0x0024	USART, RX	USART Rx Complete
20	0x0026	USART, UDRE	USART, Data Register Empty
21	0x0028	USART, TX	USART, Tx Complete
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator
25	0x0030	TWI	2-wire Serial Interface
26	0x0032	SPM READY	Store Program Memory Ready

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming" on page 279.
2. When the IVSEL bit in MCUCR is set, Interrupt Vectors will be moved to the start of the Boot Flash Section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash Section.

Interrupciones

- Para modelos con sección de arranque, la tabla de vectores o/y reset puede ubicarse o en la sección de aplicación o arranque dependiendo de unos fusibles.

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x002
1	1	0x000	Boot Reset Address + 0x0002
0	0	Boot Reset Address	0x002
0	1	Boot Reset Address	Boot Reset Address + 0x0002

- Ejemplo de inicialización de tabla de vectores.

```
Address  Labels Code                 Comments
0x0000          jmp    RESET         ; Reset Handler
0x0002          jmp    EXT_INT0      ; IRQ0 Handler
0x0004          jmp    EXT_INT1      ; IRQ1 Handler
0x0006          jmp    PCINT0        ; PCINT0 Handler
0x0008          jmp    PCINT1        ; PCINT1 Handler
0x000A          jmp    PCINT2        ; PCINT2 Handler
0x000C          jmp    WDT           ; Watchdog Timer Handler
0x000E          jmp    TIM2_COMPA    ; Timer2 Compare A Handler
0x0010          jmp    TIM2_COMPB    ; Timer2 Compare B Handler
0x0012          jmp    TIM2_OVF      ; Timer2 Overflow Handler
```

Interrupciones

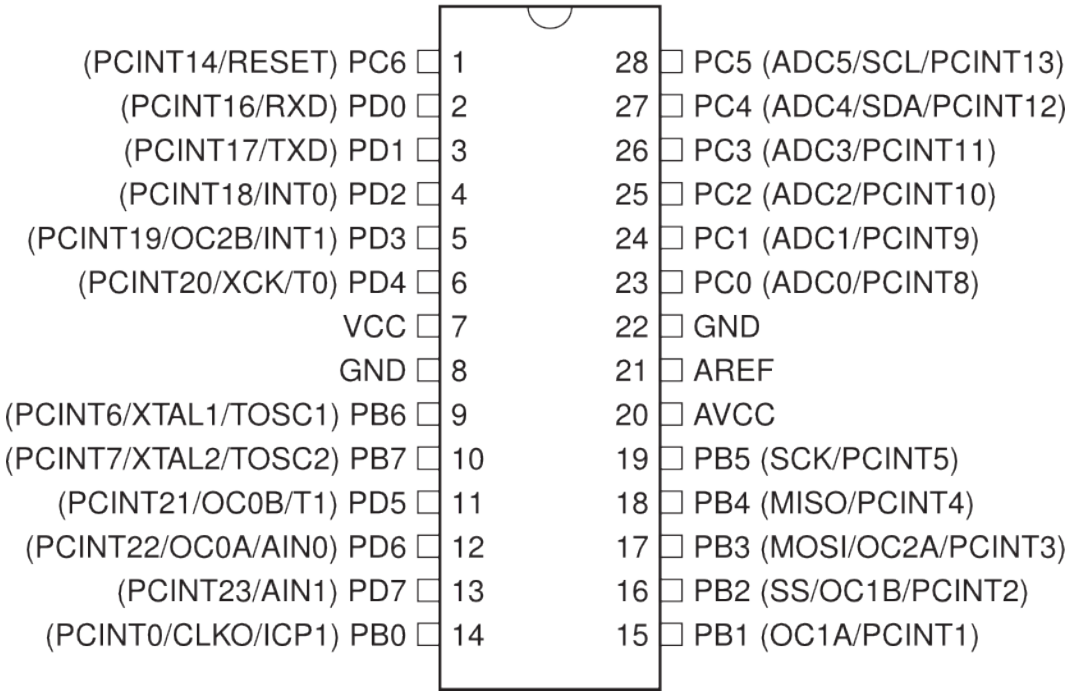
- Para que se procesen las interrupciones, el bit I del registro de estado debe estar a 1 lógico.
- Durante la ejecución de la interrupción, el Bit I se pone a 0 lo que impide el anidamiento de interrupciones.
- La rutina de interrupción debe terminar con la instrucción RETI que además pone I=1.
- En la rutina de interrupción se debe salvar, como mínimo, el registro de estado SREG. Para ello se usará la propia PILA.

```
                .CSEG
                .ORG 0
                JMP main
                JMP IRQ0_handler
                JMP IRQ_handler
                ..
main:           {Código programa principal}
                .......
IRQ0_handler:   IN r16,SREG
                PUSH r16
                {Código interrupción IRQ0}
                POP  r16
                OUT SREG,r16
                RETI
IRQ1_handler:   IN r16,SREG
                PUSH r16
                {Código interrupción IRQ1}
                POP  r16
                OUT SREG,r16
                RETI
```

Estructura de un programa con interrupciones

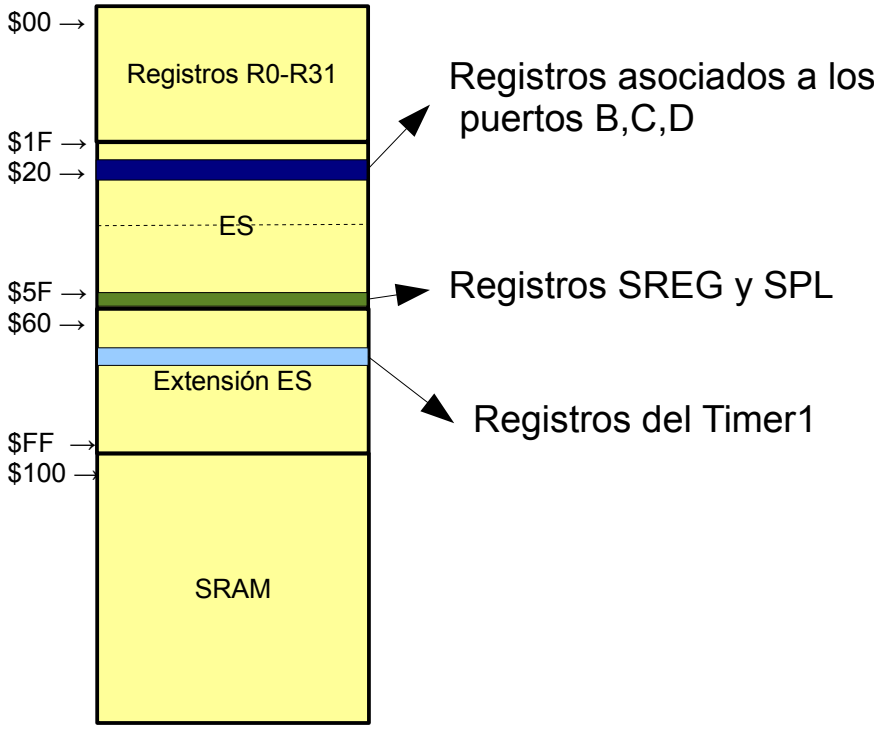
Entrada / Salida

- Puertos de entrada/salida
  - PB_{7-o}
  - PC_{7-0}
  - PD_{7-0}
- Temporizadores
  - Timer 0
  - Timer 1



Entrada / Salida

- Acceso a los puertos y al timer1
  - Puertos B, C y D
    - Zona baja de la ES
    - Permite SBI, CBI, IN, OUT, SBIC,SBIS, modos directos e indirectos.
  - Registos SREG y SPL
    - Zona alta de la ES
    - Permite IN, OUT y modos directos e indirectos
  - Timer1
    - Extensión ES
    - Sólo modos directos e indirectos



Puertos E/S

Son entradas / salidas digitales que permiten escribir valores lógicos en cada uno de los pines. Típicamente: Uno=5V, Cero=0V

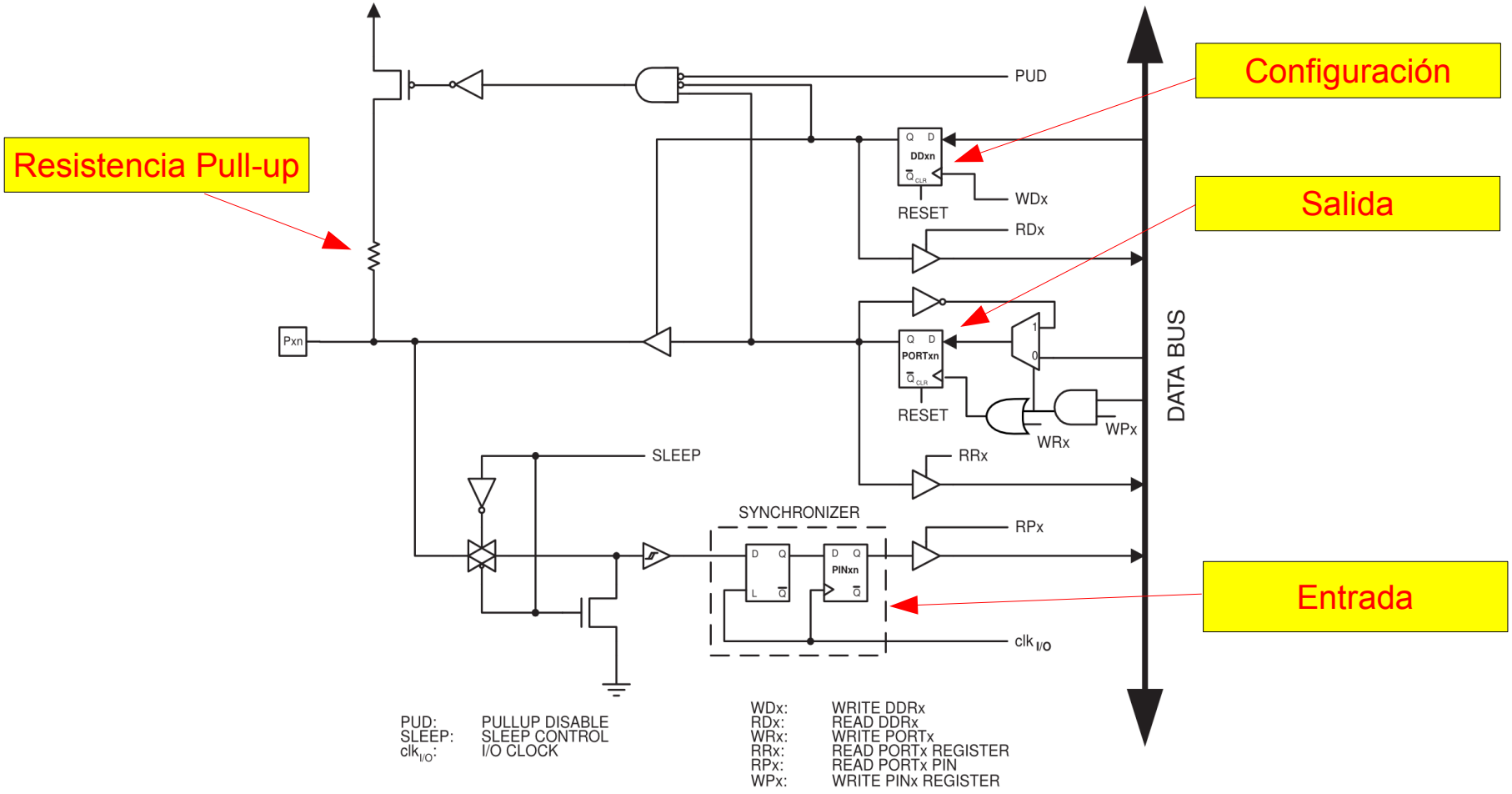
Existen tres puertos de 8 bits: Puerto B, Puerto C y Puerto D

Los puertos tienen funciones alternativas:

Ejemplo PB0: "*Capturador de eventos del temporizador*" ó "*Salida de reloj interno*" ó "*Interrupción 0 ante cambio en el PIN*"

Activar la función alternativa de un PIN no afecta al resto del pines del puerto

Puertos E/S


Resistencia Pull-up
Configuración
Salida
Entrada
PUD
DDxn
RESET
WDx
RDx
Pxn
PORTxn
RESET
WRx
WPx
DATA BUS
SLEEP
RRx
SYNCHRONIZER
RPx
PINxn
clk I/O
PUD: PULLUP DISABLE
SLEEP: SLEEP CONTROL
clk I/O: I/O CLOCK
WDx: WRITE DDRx
RDx: READ DDRx
WRx: WRITE PORTx
RRx: READ PORTx REGISTER
RPx: READ PORTx PIN
WPx: WRITE PINx REGISTER

Puertos E/S

Los pines del puertos tienen resistencias de Pull-Up que pueden activarse

Tienen un circuito de sincronización para leer los valores lógicos.

Cada puerto tiene asociado 3 registros: (x={B,C,D})

- $DDRX_{7-0}$: Configura la dirección de cada PIN (entrada o salida)
- $PORTX_{7-0}$: Registro de datos del puerto para escribir en el puerto
- $PINX_{7-0}$: Permite leer directamente en el PIN independientemente del valor $DDRX_i$

Puertos E/S

Uso del puerto B *(C y D son similares)*

Registro $DDRB_{7-0}$ (R/W): El valor del bit DDB**X** indica si el pin PB**X** es una entrada (0) o es salida (1)

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	**DDB7**	**DDB6**	**DDB5**	**DDB4**	**DDB3**	**DDB2**	**DDB1**	**DDB0**	**DDRB**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

$PINB_{7-0}$ (R) Permite la la lectura de valores lógicos de los pines

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	**PINB7**	**PINB6**	**PINB5**	**PINB4**	**PINB3**	**PINB2**	**PINB1**	**PINB0**	**PINB**
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Puertos E/S

Uso del puerto B

Registro PORTB (R/W):

Si en DDRBx está configurado como salida, el valor de PORTBx se muestra en el PIN de salida PBx

Si en DDRBx está configurado como entrada, al escribir un '1' en PORTBx activa la resistencia Pull-Up del PIN PBx

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Puertos E/S

Ejemplo 1: Establecer el puerto B como salida y activar el PINB3 a '1' y el PINB6 a '0'

```
LDI R16,0xFF
OUT DDRB,R16 ← Puerto entero como salida
SBI PORTB,3  ← Establecer el bit 3 a 1
CBI PORTB,6  ← Establecer el bit 6 a 0
```

Puertos E/S

Ejemplo 2: Establecer el PINC4 del puerto C como entrada y tomar una decisión en función del valor leído:

```
CBI  DDRC,4    ← PIN4 como entrada DDRC4=0
SBIS PIND,4    ← Esquiva una instrucción
                 si PIND4=1
JMP  PD4_ES_0  ← Salto si PIND4=0
JMP  PD4_ES_1  ← Salto si PIND4=1
```

Ejercicio: Escribir un código equivalente con la instrucción SBIC

Temporizadores

Hay 3 timers disponibles del que sólo se estudiará el

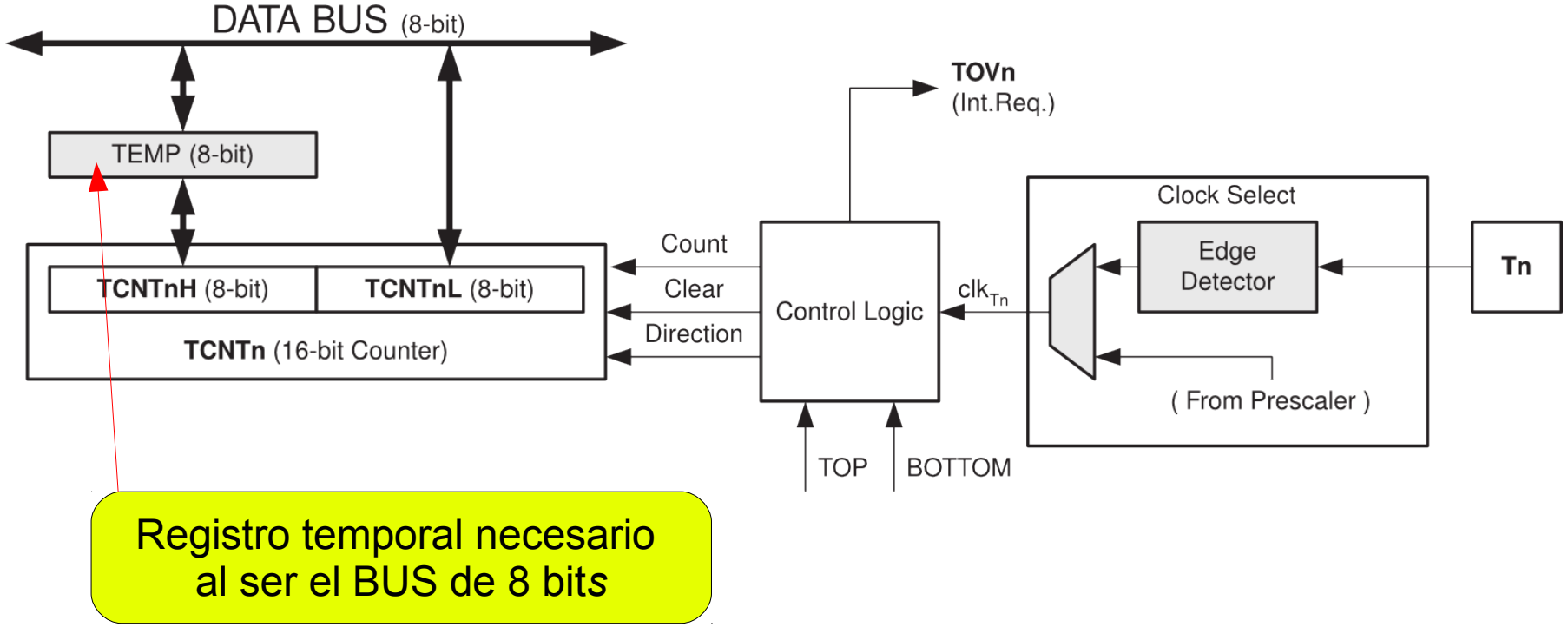
- Timer 1 de 16bit

Funcionalidad:

- Generar eventos de forma periódica
- Contador de eventos
- Generador de señales
  - PWM (Pulse Width Modulartor)
- Auto reinicio al alcanzar valores programados
- Prescaler
- Interrupciones

Temporizadores

Esquema general del temporizador 1 (16 bits)

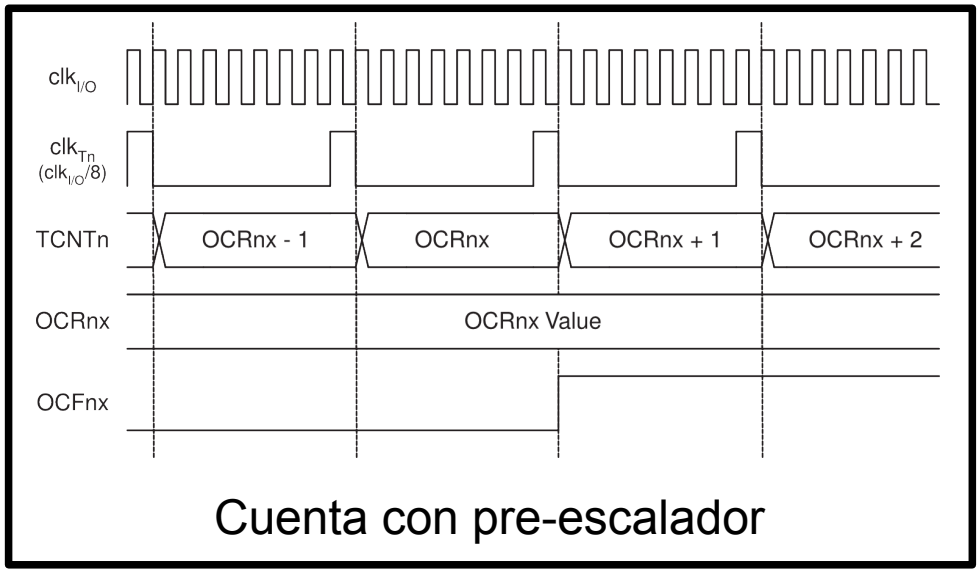


Temporizadores

Pre-escalador:

Divisor de frecuencia previo al reloj del temporizador.

Permite disminuir la frecuencia de cuenta



Cuenta con pre-escalador

Temporizadores

De los modos de operación posibles estudiaremos:

- Normal: Cuenta ascendente de manera indefinida. Después del estado de cuenta $FFFF pasa al $0000
- Puesta a cero al llegar a un valor: El contador se pone a cero automáticamente cuando se alcanza el valor establecido en OCR1A

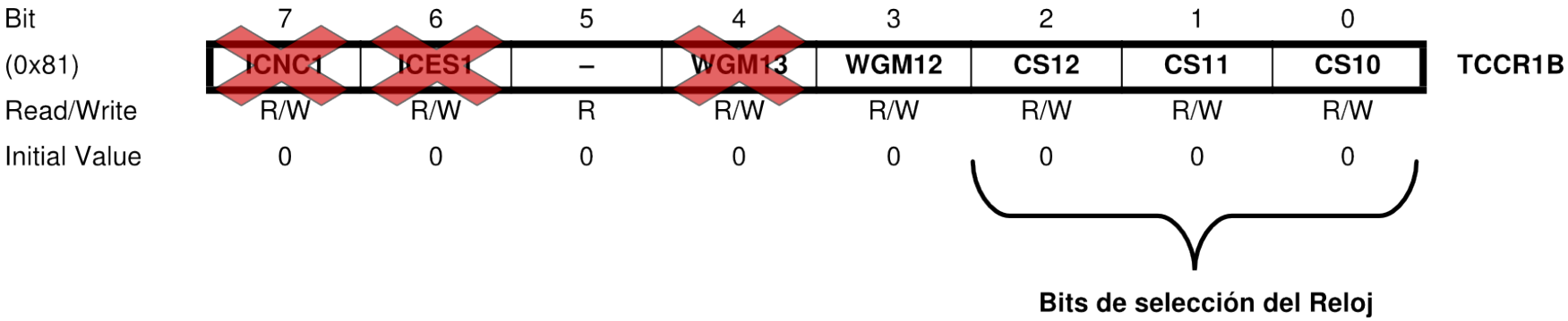
Registros involucrados

- Configuración: ~~TCCR1A~~ y TCCR1B
- Estado de cuenta: TCNT1H y TCNT1L
- Comparadores: OCR1AH ,OCR1AL, ~~OCR1BH~~ y ~~OCR1BL~~

Temporizadores

Registro de configuración:

- Solo estudiaremos 4 bits el TCCR1B
- El resto se pueden quedar sin inicializar, por defecto todos están a cero



Temporizadores

Frecuencia de funcionamiento del temporizador en función de 3 bits (CS12,CS11,CS10)

CS12	CS11	CS10	Descripción
0	0	0	Temporizador parado
0	0	1	Frecuencia clk/1
0	1	0	Frecuencia clk/8
0	1	1	Frecuencia clk/64
1	0	0	Frecuencia clk/256
1	0	1	Frecuencia clk/1024
1	1	0	Pin T1 en flanco de bajada
1	1	1	Pin T1 en flanco de subida

Temporizadores

Modos de funcionamiento:

Modo Normal:

El bit WGM12 debe configurarse a '0' (registro TCCR1B).

El contador cuenta desde 0x0000 a 0xFFFF y vuelta a 0x0000, a la frecuencia de reloj configurada.

En cada paso por 0x0000 se activa un bit llamado TOV1

Modo CTC (Clear Timer on Compare Match):

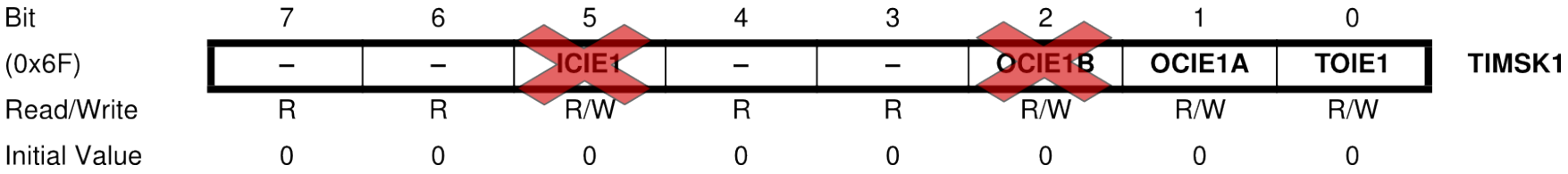
El bit WGM12 debe configurarse a '1' (registro TCCR1B)

El contador cuenta desde 0x0000 hasta que su contenido es igual al almacenado en OCR1A (16 bits). Tras esto se pone automáticamente a 0x0000.

Cuando esto ocurre se activa un bit llamado OCF1

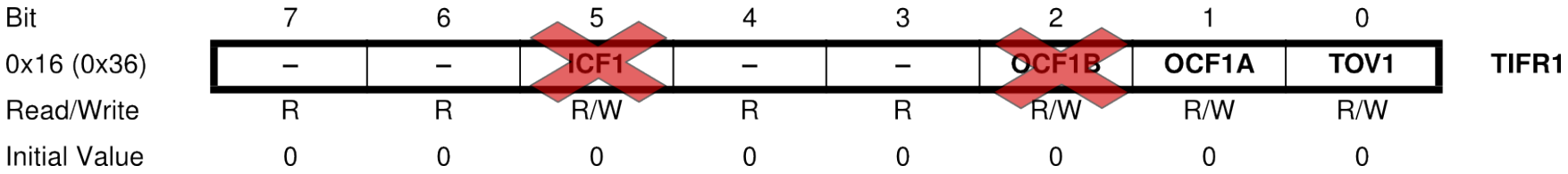
Temporizadores

Registros de habilitación de interrupciones



Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Registro de banderas de interrupciones



Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	TIFR1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Temporizadores

Interrupciones:

Si TOIE1=1 (registro TIMSK1) entonces se produce una interrupción cada vez que el temporizador pasa por 0x0000 y se activa la bandera de interrupción TOV1 (registro TIFR1)

Si OCIE1A=1 (registro TIMSK1) entonces se produce una interrupción cada vez que el temporizador alcanza el valor almacenado en OCR1A se pone a 0x0000 y se activa la bandera de interrupción OCF1A (registro TIFR1)

Si OCIE1B=1 (registro TIMSK1) entonces se produce una interrupción cada vez que el temporizador alcanza el valor almacenado en OCR1B se pone a 0x0000 y se activa la bandera de interrupción OCF1B (registro TIFR1)

Temporizadores

Importante: El contador es de 16bits y el bus de 8. La escritura de registros de 16bits se debe hacer en 2 pasos y en un orden correcto:

1º Parte alta del registro. Por ejemplo OCR1AH

2º Parte baja del registro. Por ejemplo OCR1AL

```
LDI R16,0x10
STS OCR1AH,R16 ← Escritura de la parte alta,realmente no
                 se escribe el registro, se queda en un
                 registro temporal
LDI R16,0x02
STS OCR1AL,R16 ← Se dispara escritura simultánea de 16
                 bits: 8 desde un registro temporal y 8
                 desde el bus del sistema
```

Temporizadores

Ejemplo: Con un micro con reloj a 1Mhz conseguir que el contador se reinicie 1 vez por segundo

1º Bajar la frecuencia del reloj con el preescalador: 1Mhz/64=15625Hz

2º Cargar en OCR1A el valor 15625 = $3D09

3º Activa el auto-clear cuando el contenido el temporizador sea igual a OCR1A.

Cada vez que se cicle el contador ha pasado un segundo

Temporizadores

Solución del ejemplo

```
LDI   R16,0x3D          ← Carga 0x3D09 en OCR1A
STS   OCR1AH,R16        ← Primero parte alta pero OCR1A
                          todavía queda inalterado. No se
                          puede usar OUT.
LDI   R16,0x09
STS   OCR1AL,R16        ← Tras escribir la parte baja se
                          escribe el registro completo de
                          16bits
LDI   R16,0b00001011   ← Activa el modo CLC, bit WGM12=1
                          Prescaler CLK/64
STS   TCCR1B,R16        ← A partir de esta instrucción el
                          timer está funcionando
```

Bibliografía

Instruction Set datasheet.

Atmegax8pa datasheet.

Microcontroller projects with the atmel controller. Gadre, Dhananjay.

Atmel AVR microcontroller primer: programming and interfacing. Barlett and Pack

Embedded Systems Design with the Atmel AVR Microcontroller. Steven Barret.