+-----+
| MCU | AT90S2313
| U01 | โดย อ.กำธร เรือนฝายกาศ
+-----+

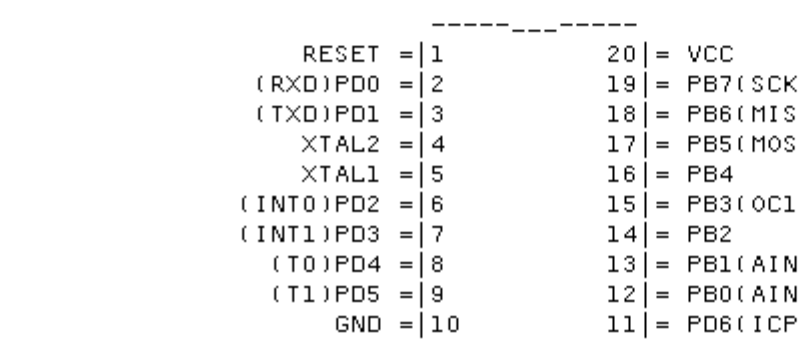
(1) กล่าวโดยทั่วไป

ไอซีไมโครคอนโทรลเลอร์(MCU - Micro-Controller Unit) ที่เราเลือกนำมาใช้ศึกษา เป็นผลิตภัณฑ์จากบริษัท Atmel(www.atmel.com) เบอร์ AT90S2313 เป็นไอซีซีมอส 20 ขา ซึ่งมีคุณสมบัติที่สำคัญดังต่อไปนี้

<> โครงสร้างเป็นแบบ AVR RISC
<> ประสิทธิภาพสูง และใช้กำลังงานต่ำ
- มี 118 คำสั่ง ส่วนมากทำงานเสร็จใน 1 cycle
- รีจิสเตอร์ 8 บิต จำนวน 32 ตัว
- ความเร็ว 10 MIPS ที่ความถี่ 10MHz
<> หน่วยความจำ มี 3 ส่วนคือ
- flash memory จำนวน 1K words (2K bytes) เขียน/ลบ ได้ไม่ต่ำกว่า 1000 ครั้ง
- SRAM 128 ไบต์
- EEPROM 128 ไบต์ เขียน/ลบ ได้ไม่ต่ำกว่า 100000 ครั้ง
- มีระบบรักษาความปลอดภัย สามารถล็อคไม่ให้อ่านโปรแกรมใน flash memory และข้อมูลใน EEPROM ได้
<> อุปกรณ์ภายใน
- มี Timer/counter 8 บิต จำนวน 1 ตัว พร้อมวงจรหาร ความถี่คล็อคแยกอิสระ
- มี Timer/conter 16 บิต จำนวน 1 ตัว พร้อมวงจรหาร ความถี่คล็อคแยกอิสระ,วงจรเปรียบเทียบ,Capture mode และ PWM 8,9 หรือ 10บิต
- อะนาลอกคอมพาราเตอร์ จำนวน 1 ตัว
- Programmable Watchdog Timer พร้อม on-chip Oscillator 1 ชุด
- SPI สำหรับทำ In-System Programming
- พอร์ตอนุกรม Full duplex UART 1 ชุด
<> คุณสมบัติพิเศษ
- Low power Idle and power down mode
- External and Internal Ineyrupt Source
- เป็นไอซีแบบซีมอสความเร็วสูง
- แรงดันใช้งาน 2.7-6V สำหรับ (AT90S2313-4) และ 4-6V สำหรับ AT90S2313-10
- ความถี่คล็อค 0-4MHz สำหรับ AT90S2313-4 และ 0-10MHz สำหรับ AT90S2313-10
- ที่ความถี่ 4MHz แรงดัน 3 โวลต์ อุณหภูมิ 25'C จะใช้กระแสไฟฟ้าในสภาวะ Active เท่ากับ 2.8mA ในสภาวะ Idle เท่ากับ 0.8mA และในสภาวะ Power Down จะใช้กระแสไฟฟ้าน้อยกว่า 1uA

(2) การจัดขา

*** ตัวถังแบบ PDIP/SOIC



<> PORTB (PB7..PB0)
เป็นพอร์ต 8 บิต สามารถโปรแกรมให้แต่ละบิตของพอร์ตเป็นอินพุตหรือเอ้าต์พุตก็ได้แยกกันอิสระในแต่ละบิต แต่ละขาของพอร์ตสามารถจ่ายกระแสออกไปให้กับอุปกรณ์ภายนอกเช่น LED ได้ถึง 20mA นอกจากนี้ขา PB0 และ PB1 ยังถูกใช้เป็นอินพุต AIN0 และ AIN1 สำหรับ analog comparator อีกด้วย

<> PORTD
เป็นพอร์ต 7 บิต คุณสมบัติคล้ายกันกับ PORTB นอกจากนี้ PORTD ยังใช้สำหรับทำหน้าที่อื่นได้อีกคือ
PD0 = RXD เป็นขาอินพุตสำหรับพอร์ตอนุกรม
PD1 = TXD เป็นขาเอ้าต์พุตสำหรับพอร์ตอนุกรม
PD2 = INT0 เป็นขารับอินเตอร์รัพท์จากภายนอก
PD3 = INT1 เป็นขารับอินเตอร์รัพท์จากภายนอก
PD4 = T0 เป็นขาสำหรับรับสัญญาณ clock จากภายนอกสำหรับ Timer/counter0
PD5 = T1 เป็นขาสำหรับรับสัญญาณ clock จากภายนอกสำหรับ Timer/counter1
PD6 = ICP ใช้เป็นอินพุตสำหรับสั่ง Timer/counter1 เก็บค่าที่กำลังนับไว้ในรีจิสเตอร์(ดูตาต้าชีทหน้า 32 เพิ่มเติม)

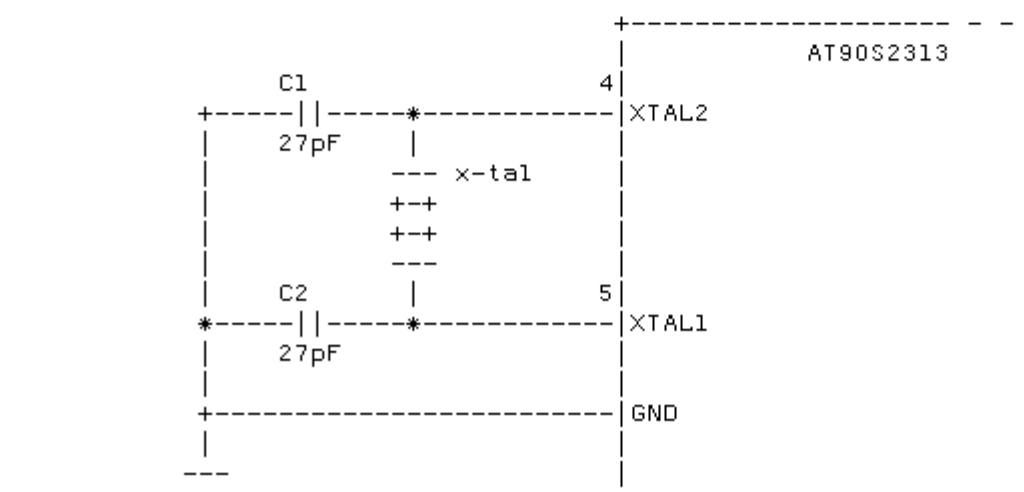
<> RESET
เป็นขาสำหรับรีเซ็ท ลอจิก 0 ที่มีเวลามากกว่า 50ns จะทำให้เกิดการรีเซ็ท แม้นว่าจะไม่มีสัญญาณคล็อคก็ตาม

<> XTAL1
เป็นขาอินพุตของ inverting oscillator amplifier สำหรับรับสัญญาณคล็อคจากภายนอก และเป็นอินพุตสำหรับการทำงานของคล็อคภายใน(กรณีที่เราต่อ x-tal ไว้)

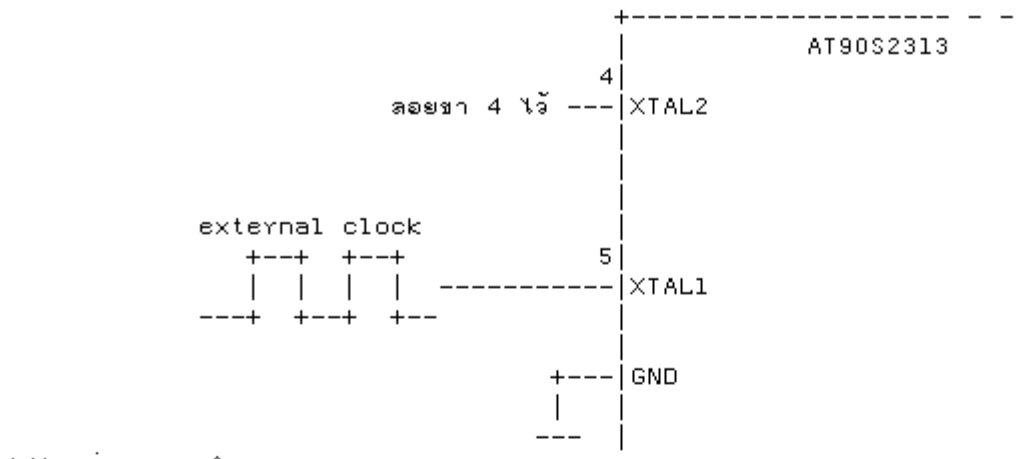
<> XTAL0
เป็นขาเอ้าต์พุตของ inverting oscillator amplifier

(3) วงจร Crystal Oscillator

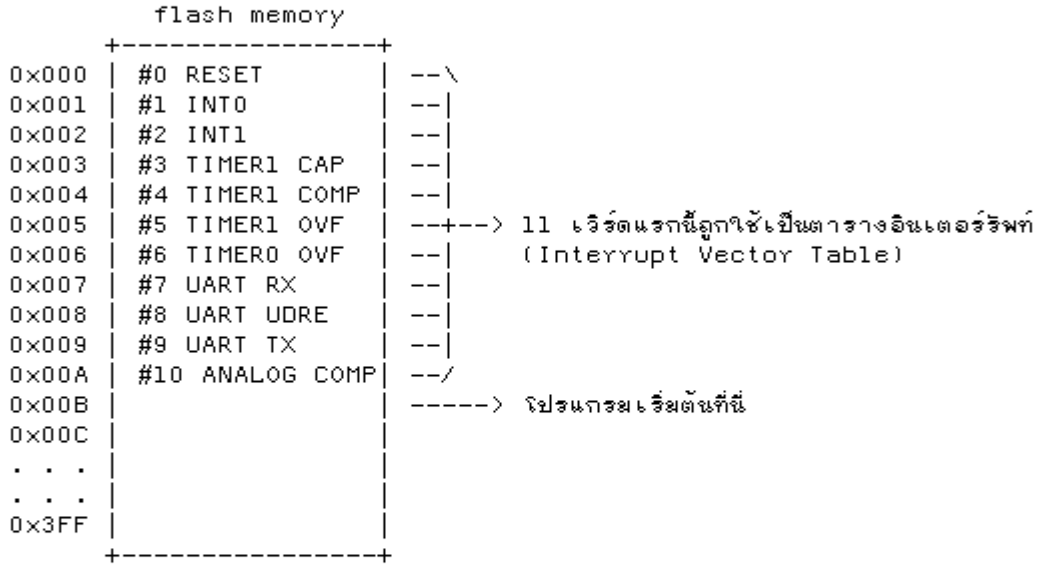
ในกรณีที่เราใช้คล็อคภายใน เราจะต้องต่อคริสตอลเข้ากับขา 4 และขา 5 พร้อมกับคาปาซิเตอร์ 27pF อีก 2 ตัว ความถี่ของคริสตอลนั้นขึ้นอยู่กับเบอร์ของ MCU ถ้าเป็นเบอร์ AT90S2313-4 ก็จะใช้คริสตอลที่มีความถี่ได้ไม่เกิน 4MHz และไม่เกิน 10MHz สำหรับเบอร์ AT90S2313-10



ถ้าต้องการใช้คล็อคจากภายนอก ก็ทำได้โดยลอยขา 4 ไว้ แล้วป้อนสัญญาณคล็อคจากเครื่องกำเนิดสัญญาณภายนอกเข้าไปทางขา 5



(4) หน่วยความจำ



<> flash memory

เป็นหน่วยความจำสำหรับเก็บโปรแกรม มีจำนวน 1 กิโลเวิร์ด 11เวิร์ดแรกถูกจองไว้สำหรับเป็นตารางอินเตอร์รัพท สำหรับอินเตอร์รัพท์ 11 หมายเลข (รายละเอียดของอินเตอร์รัพท์แต่ละหมายเลขดูได้จากตาต้าชีทหน้า 19)ตำแหน่งเริ่มต้นของโปรแกรมเท่ากับแอดเดรส 0x00B ไปจนถึง 0x3FF โดยแต่ละแอดเดรสของ flash memory สามารถเก็บได้ 16 บิต การเข้าถึงข้อมูล(ข้อมูลไม่ใช่โปรแกรม)ใน flash memory ทำได้โดยใช้รีจิสเตอร์ Z เป็นตัวชี้แอดเดรส และใช้คำสั่ง lpm ในการดึงเอาข้อมูลออกมา หน่วยความจำส่วนนี้ข้อมูลจะไม่หายไป แม้นว่า MCU จะไม่ได้รับไฟเลี้ยงก็ตาม จึงเหมาะที่จะใช้เก็บโปรแกรม และข้อมูลที่ตายตัว

<> SRAM

หน่วยความจำ SRAM ถูกแบ่งเป็นส่วนๆโดยส่วนแรกใช้สำหรับรีจิสเตอร์ R0 ถึง R31 จำนวน 32 ไบต์ ส่วนที่สองสำหรับรีจิสเตอร์ไอโออีก 64 ไบต์ และส่วนที่สามสำหรับให้เราใช้เขียน/อ่าน ข้อมูล จำนวน 128 ไบต์ (128 x 8) ซึ่งเหมาะที่จะใช้สำหรับเก็บข้อมูลที่ต้องการเขียน/อ่าน บ่อยๆ,มีความเร็วในการเข้าถึงข้อมูลสูง แต่ถ้าไฟดับข้อมูลจะหายไปทั้งหมด การเข้าถึงข้อมูลสามารถอ้างแอดเดรสผ่านทางรีจิสเตอร์ X,Y และ Z

<> EEPROM

เป็นหน่วยความจำขนาด 128 ไบต์ 8 บิต สามารถเขียน/อ่านได้ มีความเร็วในการเข้าถึงข้อมูลต่ำ(มากกว่า 2.5mS ต่อหนึ่งไบต์ ที่เขียน) ดังนั้นจึงเหมาะที่จะใช้เก็บข้อมูลที่ไม่ต้องการความเร็ว สูงมากนัก และเนื่องจากเป็นหน่วยความจำที่ไฟดับแล้วข้อมูลไม่หายไป จึงเหมาะที่จะนำไปใช้เก็บข้อมูลที่มีการเปลี่ยนแปลงโดยผู้ใช้ เช่น ถ้าเรานำเอา MCU ไปทำเป็นเทอร์โมสตัท เราก็นำอุณหภูมิที่ตั้งไว้ เพื่อให้เทอร์โมสตัทต่อหรือตัดวงจร มาเก็บไว้ในหน่วยความจำส่วนนี้(รายละเอียดเพิ่มเติมดูได้จากดาต้าชีทของไอซีหน้า 38)

(5) ตระกูล AVR Microcontroller

นอกจากไอซีเบอร์ AT90S2313 แล้วยังมีเบอร์อื่นๆ อีกที่มีความสามารถต่ำหรือสูงกว่านี้ มีขาตั้งแต่ 6 ขาไปจนถึง 40 ขา เช่น AT90S2323/2343 ,AT90S90S8515/4414/8535/4434 เป็นต้น เราสามารถเข้าไปดูข้อมูลดาต้าชีท ตัวอย่างการประยุกต์ใช้งาน ได้จากเวบไซท์ของบริษัท Atmel

(6) Instruction mnemonic

ตัวแปรภาษา Assembler สำหรับ AVR ยอมรับคำสั่ง mnemonic ดังต่อไปนี้
(รายละเอียดเพิ่มเติมดูได้จาก AVR Data Book

ARITHMETIC AND LOGIC INSTRUCTIONS

Mne.	Operand	Description	Operation	Flags	#Clock
ADD	Rd, Rr	Add without Carry	Rd<-Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry	Rd<-Rd + Rr + C	Z,C,N,V,H	1
SUB	Rd, Rr	Subtract without Carry	Rd<-Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Immediate	Rd<-Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry	Rd<-Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract Immediate with carry	Rd<-Rd - K - C	Z,C,N,V,H	1
AND	Rd, Rr	Logical AND	Rd<-Rd & Rr	Z,N,V	1
ANDI	Rd, K	Logical AND with immediate	Rd<-Rd & K	Z,N,V	1
OR	Rd, Rr	Logical OR	Rd<-Rd | Rr	Z,N,V	1
ORI	Rd, K	Logical OR with immediate	Rd<-Rd | K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR	Rd<-Rd ^ Rr	Z,N,V	1
COM	Rd	Ones Complement	Rd<-0xFF - Rd	Z,C,N,V	1
NEG	Rd	Twos Complement	Rd<-0x00 - Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd<-Rd | K	Z,N,V	1
CBR	Rd,K	Clear Bit in Register	Rd<-Rd&(0xFF-K)	Z,N,V	1
INC	Rd	Increment	Rd<-Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd<-Rd - 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd<-Rd & Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd<-Rd ^ Rd	Z,N,V	1
SER	Rd	Set Register	Rd<-0xFF	None	1
ADIW	Rdl, K	Add Immediate to Word	Rdh:Rdl<-Rdh:Rdl+K	None	1
SBIW	Rdl, K	Subtract Immediate from Word	Rdh:Rdl<-Rdh:Rdl-K	None	1
MUL	Rd,Rr	Multiply Unsigned	R1,R0<-Rd x Rr	C	2*

BRANCH INSTRUCTIONS

Mne.	Operand	Description	Operation	Flags	#Clock
RJMP	k	Relative Jump	PC<-PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC<-Z	None	2
JMP	k	Jump	PC<-k	None	3
RCALL	k	RelativeCall Subroutine	PC<-PC + k + 1	None	3
ICALL		IndirectCall to (Z)	PC<-Z	None	3
CALL	k	Call Subroutine	PC<-k	None	4
RET		Subroutine Return	PC<-STACK	None	4
RETI		Interrupt Return	PC<-STACK	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if(Rd=Rr) PC<-PC+2 or 3	None	1/2
CP	Rd,Rr	Compare	Rd - Rr	Z,C,N,V,H	1
CPC	Rd,Rr	Compare with Carry	Rd - Rr - C	Z,C,N,V,H	1
CPI	Rd,K	Compare with Immediate	Rd - K	Z,C,N,V,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if(Rr(b)=0) PC<-PC+2 or 3	None	1/2
SBRS	Rr, b	Skip if Bit in Register Set	if(Rr(b)=1) PC<-PC+2 or 3	None	1/2
SBIC	P, b	Skip if Bit in I/O Register Cleared	if(I/O(P,b)=0) PC<-PC+2 or 3	None	2/3
SBIS	P, b	Skip if Bit in I/O Register Set	If(I/O(P,b)=1) PC<-PC+2 or 3	None	2/3
BRBS	s, k	Branch if Status Flag Set	if(SREG(s)=1) PC<-PC+k+1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if(SREG(s)=0) PC<-PC+k+1	None	1/2
BREQ	k	Branch if Equal	if(Z=1) PC<-PC+k+1	None	1/2
BRNE	k	Branch if Not Equal	if(Z=0) PC<-PC+k+1	None	1/2
BRCS	k	Branch if Carry Set	if(C=1) PC<-PC+k+1	None	1/2
BRCC	k	Branch if Carry Cleared	if(C=0) PC<-PC+k+1	None	1/2
BRSH	k	Branch if Same or Higher	if(C=0) PC<-PC+k+1	None	1/2
BRLO	k	Branch if Lower	if(C=1) PC<-PC+k+1	None	1/2
BRMI	k	Branch if Minus	if(N=1) PC<-PC+k+1	None	1/2
BRPL	k	Branch if Plus	if(N=0) PC<-PC+k+1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if(N V=0) PC<-PC+k+1	None	1/2
BRLT	k	Branch if Less Than, Signed	if(N V= 1) PC<-PC+k+1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if(H=1) PC<-PC+k+1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if(H=0) PC<-PC+k+1	None	1/2
BRTS	k	Branch if T Flag Set	if(T=1) PC<-PC+k+1	None	1/2
BRTC	k	Branch if T Flag Cleared	if(T=0) PC<-PC+k+1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if(V=1) PC<-PC+k+1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if(V=0) PC<-PC+k+1	None	1/2
BRIE	k	Branch if Interrupt Enabled	if(I=1) PC<-PC+k+1	None	1/2
BRID	k	Branch if Interrupt Disabled	if(I=0) PC<-PC+k+1	None	1/2

DATA TRANSFER INSTRUCTIONS

Mne.	Oparand	Description	Operation	Flags	#Clock
MOV	Rd, Rr	Copy Register	Rd<-Rr	None	1
LDI	Rd, K	Load Immediate	Rd<-K	None	1
LDS	Rd, k	Load Direct	Rd<-(k)	None	3
LD	Rd, X	Load Indirect	Rd<-(X)	None	2
LD	Rd, X+	Load Indirect and Post-Increment	Rd<-(X), X<-X+1	None	2
LD	Rd, -X	Load Indirect and Pre-Decrement	X<-X-1, Rd<-(X)	None	2
LD	Rd, Y	Load Indirect	Rd<-(Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Increment	Rd<-(Y), Y<-Y+1	None	2
LD	Rd, -Y	Load Indirect and Pre-Decrement	Y<-Y-1, Rd<-(Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd<-(Y+q)	None	2
LD	Rd, Z	Load Indirect	Rd<-(Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Increment	Rd<-(Z), Z<-Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Decrement	Z<-Z-1, Rd<-(Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd<-(Z+q)	None	2
STS	k, Rr	Store Direct	(k)<-Rr	None	3
ST	X, Rr	Store Indirect	(X)<-Rr	None	2
ST	X+, Rr	Store Indirect and Post-Increment	(X)<-Rr, X<-X+1	None	2
ST	-X, Rr	Store Indirect and Pre-Decrement	X<-X-1, (X)<-Rr	None	2
ST	Y, Rr	Store Indirect and Pre-Decrement	(Y)<-Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Increment	(Y)<-Rr, Y<-Y+1	None	2
ST	-Y, Rr	Store Indirect and Pre-Decrement	Y<-Y-1, (Y)<-Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y+q)<-Rr	None	2
ST	Z, Rr	Store Indirect	(Z)<-Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Increment	(Z)<-Rr, Z<-Z+1	None	2
ST	-Z, Rr	Store Indirect and Pre-Decrement	Z<-Z-1, (Z)<-Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z+q) <- Rr	None	2
LPM		Load Program Memory	R0<-(Z)	None	3
IN	Rd, P	In Port	Rd<-P	None	1
OUT	P, Rr	Out Port	P<-Rr	None	1
PUSH	Rr	Push Register on Stack	STACK <- Rr	None	2
POP	Rd	Pop Register from Stack	Rd <- STACK	None	2

BIT AND BIT-TEST INSTRUCTIONS

Mne.	Operand	Description	Operation	Flags	#Clock
LSL	Rd	Logical Shift Left	Rd(n+1)<-Rd(n) Rd(0)<-0 C<-Rd(7)	Z,C,N,V,H	1
LSR	Rd	Logical Shift Right	Rd(n)<-Rd(n+1) Rd(7)<-0 C<-Rd(0)	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)<-C Rd(n+1)<-Rd(n) C<-Rd(7)	Z,C,N,V,H	1
ROR	Rd	Rotate Right Through Carry	Rd(7)<-C Rd(n)<-Rd(n+1) C<-Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n)<-Rd(n+1) n=0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0)<-->Rd(7..4)	None	1
BSET	s	Flag Set	SREG(s) <- 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) <- 0	SREG(s)	1
SBI	P, b	Set Bit in I/O Register	I/O(P, b) <- 1	None	2
CBI	P, b	Clear Bit in I/O Register	I/O(P, b) <- 0	None	2
BST	Rr, b	Bit Store from Register to T	T <- Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) <- T	None	1
SEC		Set Carry	C <- 1	C	1
CLC		Clear Carry	C <- 0	C	1
SEN		Set Negative Flag	N <- 1	N	1
CLN		Clear Negative Flag	N <- 0	N	1
SEZ		Set Zero Flag	Z <- 1	Z	1
CLZ		Clear Zero Flag	Z <- 0	Z	1
SEI		Global Interrupt Enable	I <- 1	I	1
CLI		Global Interrupt Disable	I <- 0	I	1
SES		Set Signed Test Flag	S <- 1	S	1
CLS		Clear Signed Test Flag	S <- 0	S	1
SEV		Set Twos Complement Overflow	V <- 1	V	1
CLV		Clear Twos Complement Overflow	V <- 0	V	1
SET		Set T in SREG	T <- 1	T	1
CLT		Clear T in SREG	T <- 0	T	1
SEH		Set Half Carry Flag in SREG	H <- 1	H	1
CLH		Clear Half Carry Flag in SREG	H <- 0	H	1
NOP		No Operation		None	1
SLEEP		Sleep(see specific descr. for Sleep)		None	1
WDR		Watchdog Reset(see specific descr. for WDR)		None	1

หมายเหตุ
=======
- คำสั่งที่ทำเครื่องหมาย * ไม่มีให้ใช้ในเบอร์ AT90S2313
- ตัวอักษรตัวพิมพ์ใหญ่หรือเล็กไม่มีผลต่อการ Assemble
- ตัว operands มีรูปแบบดังนี้

Rd	:	R0-R31 or R16-R31 (depending on instruction)
Rr	:	R0-R31
b	:	Constant (0-7), can be a constant expression
s	:	Constant (0-7), can be a constant expression
P	:	Constant (0-31/63), can be a constant expression
K	:	Constant (0-255), can be a constant expression
k	:	Constant, value range depending on instruction. Can be a constan
q	:	Constant (0-63), can be a constant expression
Rdl	:	R24, R26, R28, R30. For ADIW and SBIW instructions

- สัญญลักษณ์ทางลอจิก มีรูปแบบดังนี้

&	:	AND
|	:	OR
^	:	Exclusive OR

- คำสั่งที่ลงท้ายด้วยตัว I เช่น LDI มักจะต้องใช้กับรีจิสเตอร์ R16 ขึ้นไป
  เช่น LDI R0,7 ไม่ได้,แต่ LDI R16,7 ได้

SET	Set T in SREG	T <- 1	T	1
CLT	Clear T in SREG	T <- 0	T	1
SEH	Set Half Carry Flag in SREG	H <- 1	H	1
CLH	Clear Half Carry Flag in SREG	H <- 0	H	1
NOP	No Operation		None	1
SLEEP	Sleep(see specific descr. for Sleep)		None	1
WDR	Watchdog Reset(see specific descr. for WDR)		None	1

หมายเหตุ

- คำสั่งที่ทำเครื่องหมาย * ไม่มีให้ใช้ในเบอร์ AT90S2313
- ตัวอักษรตัวพิมพ์ใหญ่หรือเล็กไม่มีผลต่อการ Assemble
- ตัว operands มีรูปแบบดังนี้

Rd	:	R0-R31 or R16-R31 (depending on instruction)
Rr	:	R0-R31
b	:	Constant (0-7), can be a constant expression
s	:	Constant (0-7), can be a constant expression
P	:	Constant (0-31/63), can be a constant expression
K	:	Constant (0-255), can be a constant expression
k	:	Constant, value range depending on instruction. Can be a constan
q	:	Constant (0-63), can be a constant expression
Rdl	:	R24, R26, R28, R30. For ADIW and SBIW instructions

- สัญญลักษณ์ทางลอจิก มีรูปแบบดังนี้

  & : AND

  | : OR

  ^ : Exclusive OR

- คำสั่งที่ลงท้ายด้วยตัว I เช่น LDI มักจะต้องใช้กับรีจีสเตอร์ R16 ขึ้นไป
  เช่น LDI R0,7 ไม่ได้,แต่ LDI R16,7 ได้