ข้อมูลไมโครคอนโทรลเลอร์ AVR

โดย : ธวัช ไชยวรรณ์,ธนาสิทธิ นามหาญ
อ.กำธร เรือนฝ่ายกาศ
แผนกอิเล็กทรอนิกส์ สถาบันเทคโนโลยีราชมงคล วิทยาเขตภาคพายัพ
สงวนลิขสิทธิ์ พ.ศ.2545
เอกสารสำหรับแจกฟรี ห้ามจำหน่าย

คุณสมบัติ และขาต่อใช้งาน ของไมโครคอนโทรลเลอร์

คุณสมบัติ

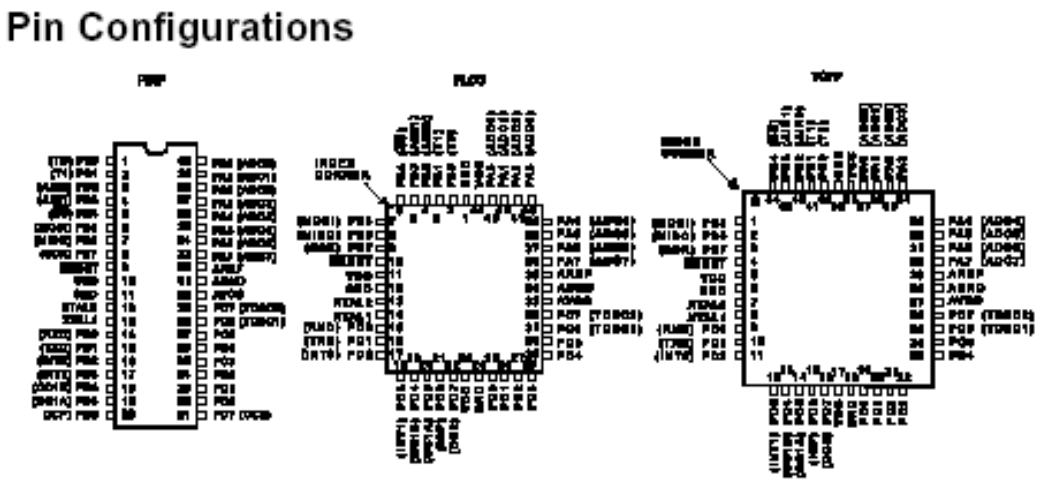
สถาปัตยกรรมภายถูกออกแบบให้ใช้สถาปัตยกรรมแบบ RISC (Reduce Instruction Set Computer)
มีคำสั่งในการควบคุมการทำงานของไมโครคอนโทรลเลอร์จำนวน 118 คำสั่ง
หน่วยความจำแบบ FLASH สำหรับบันทึก PROGRAM MEMORY ขนาด 8 Kbyte
หน่วยความจำแบบ EEPROM สำหรับบันทึก DATA MEMORY ขนาด 512 Byte
หน่วยความจำแบบ RAM ขนาด 512 Byte
ระบบการเปลี่ยนสัญญาณ ANALOG TO DIGITAL ขนาด 10 บิท จำนวน 8 CHANNEL
กลุ่มรีจิสเตอร์ใช้งานทั่วไป ขนาด 8 บิท จำนวน 32 ตัว
พอร์ตอินพุตและเอาท์พุต ขนาด 8 บิท จำนวน 4 พอร์ต
ระบบการสื่อสารข้อมูลดิจิตอลแบบอะซิงโครนัส (UART) 1 CHANNEL
ระบบการสื่อสารข้อมูลดิจิตอลแบบซิงโครนัส (SPI) 1 CHANNEL
ความถี่สัญญาณนาฬิกา 0 – 8 MHz สำหรับ AT90S/LS8535
ระบบการรีเซ็ตแบบอัตโนมัติเมื่อเริ่มจ่ายกระแสไฟฟ้าเข้าไมโครคอนโทรลเลอร์ (Power on Reset)
ระบบการกำเนิดความถี่สัญญาณแบบ PWM จำนวน 3 CHANNEL
ระบบการตรวจจับระดับสัญญาณอนาลอก (Analog Comparator)
3 SLEEP MODE : IDEL, POWER SAVE and POWER DOWN
ระบบการป้องกันการ COPY ข้อมูลภายหน่วยความจำ (LOCK FOR SOLFWARE SECURITY)
ระบบตรวจจับการทำงานผิดพลาดของ CPU (WATCHDOG TIME WITH ON – CHIP OSCILATOR)
ระบบการอินเตอร์รัพท์จากภายนอก (EXTERNAL INTERRUPT)
TIME/COUNTER ขนาด 16 บิท CHANNEL และ ขนาด 8 บิท 2 CHANNEL
Vcc : 4.0 – 6.0 V.

2

รายละเอียด

AT90S4434/AT90S8535 เป็นไมโครคอนโทรลเลอร์ขนาด 8 บิท ที่มีสถาปัตยกรรม RISC (Reduce Instruction Set Computer) ซึ่งทำให้การประมวลผลมีความเร็ว 1 คำสั่ง/1 Clock หรือ CPU สามารถประมวลคำสั่งได้ 1 MIPS/MHz

โครงสร้างและตำแหน่งขา



ภาพที่ 2.13 ตำแหน่งขา

ภายในประกอบด้วยรีจิสเตอร์ใช้งานทั่วไปขนาด 8 บิท จำนวน 32 ตัว ซึ่งแต่ละตัวจะต่อเข้ากับ ALU โดยตรง ทำให้การประมวลต่อ 1 คำสั่งมีความเร็วกว่า CPU ที่มีสถาปัตยกรรมแบบ CISC

โครงสร้างภายใน AT90S8535 จะมีหน่วยความจำสำหรับ PROGRAM MEMORY แบบ FLASH ขนาด 8 Kbyte หน่วยความจำสำหรับ DATA MEMORY แบบ EEPROM ขนาด 512 Byte และหน่วยความจำแบบ RAM ขนาด 512 Byte มีพอร์ตที่สามารถทำงานได้ 2 ทิศทาง จำนวน 32 เส้นสัญญาณ และระบบ TIMER/COUNTER จำนวน 3 ชุดที่โหมดการทำงานเสริมในส่วนของการสร้างสัญญาณ PWM และส่วยของการตรวจจับสัญญาณ Input Capture มีอุปกรณ์สื่อสารข้อมูลอนุกรม แบบ UART และ SPI และยังเสริมด้วยระบบการแปลงสัญญาณ Analog to digital ขนาด 10 บิท จำนวน 8 ช่องสัญญาณที่มีพร้อมกับ MCU มี Watchdog timer เพื่อตรวจสอบการทำงานของไมโครคอนโทรลเลอร์ และมีระบบการประหยัดพลังงานอีก 3 ระบบ

ตารางที่ 2.4 รายละเอียดของขาสัญญาณ

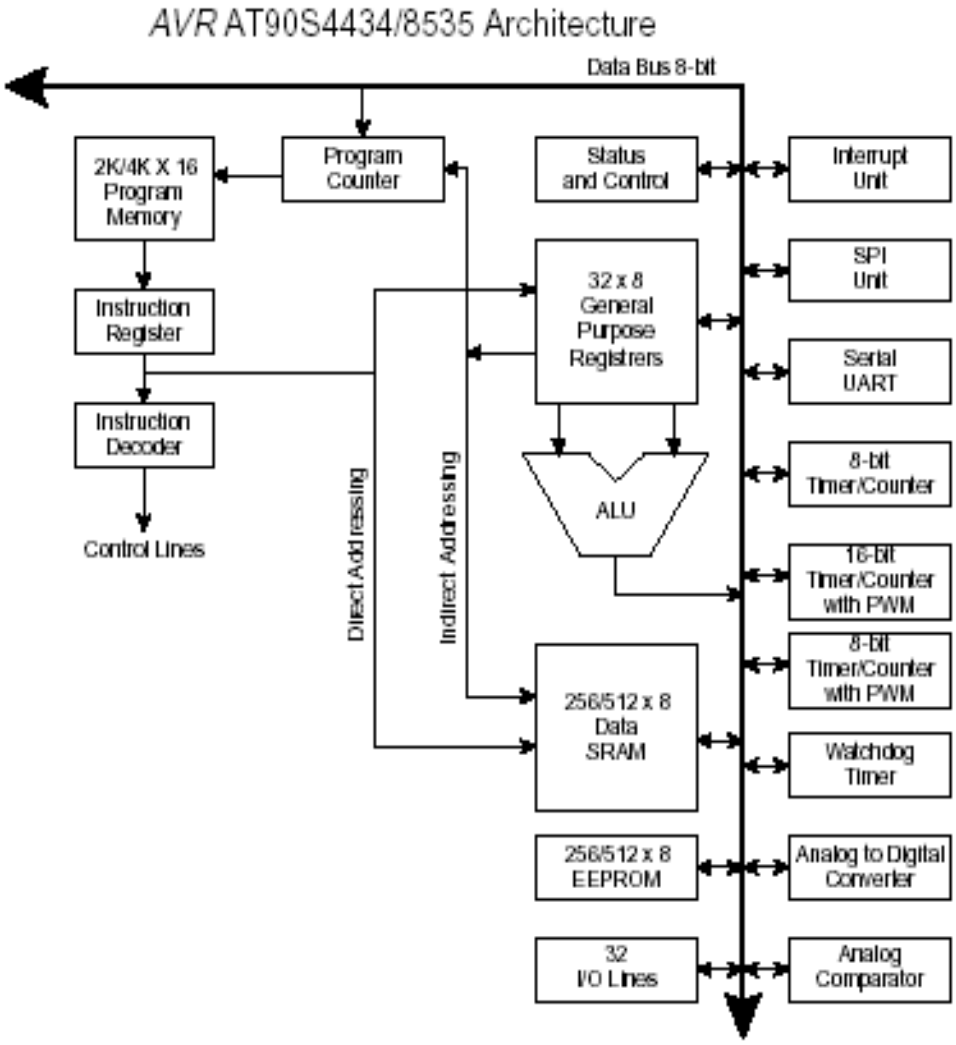
ขา	รายละเอียด
Vcc	ขาจ่ายไฟให้กับ CPU
GND	กราวด์
Port A (PA7..PA0)	เป็นพอร์ต 2 ทิศทาง ขนาด 8 บิท โดยสามารถกำหนดให้แต่ละขาของพอร์ตสามารถ PULL UP ภายในแยกจากกัน ซึ่งสามารถรับกระแส SINK 20 mA โดยพอร์ต A ยังใช้เป็นขาอินพุตเพื่อรับสัญญาณอนาลอกในส่วนของของการแปลงสัญญาณ ANALOG TO DIGITAL
Port B (PB7..PB0) Port C (PC7..PC0) Port D (PD7..PD0)	เป็นพอร์ต 2 ทิศทาง ขนาด 8 บิท โดยสามารถกำหนดให้แต่ละขาของพอร์ตสามารถ PULL UP ภายในแยกจากกัน ซึ่งสามารถรับกระแส SINK 20 mA โดยในแต่ละขาสัญญาณจะถูกนำไปใช้งานฟังก์ชั่นอื่นๆ อีก
RESET	ขารีเซ็ต
XTAL1	ขาอินพุตของ OSC
XTAL2	ขาเอาท์พุตของ OSC
Avcc	ใช้จ่ายไฟให้กับวงจร Analog to Digital ที่อยู่ภายใน MCU
AREF	เป็นขาแรงดันอ้างอิงที่ใช้งานในส่วนของวงจร Analog to Digital
AGND	ขากราวด์ของวงจร Analog to Digital

สถาปัตยกรรมภายใน และรีจิสเตอร์ใช้งานทั่วไป

สถาปัตยกรรมภายใน

โครงสร้างภายในประกอบด้วยรีจิสเตอร์ใช้งานทั่วไปขนาด 8 บิท จำนวน 32 ตัว ที่สามารถเข้าถึงข้อมูลได้ใน 1 Clock ซึ่งหมายความว่า MCU สามารถจัดการข้อมูลภายในรีจิสเตอร์ใช้งานทั่วไปได้เสร็จภายใน 1 Clock ของสัญญาณนาฬิกา

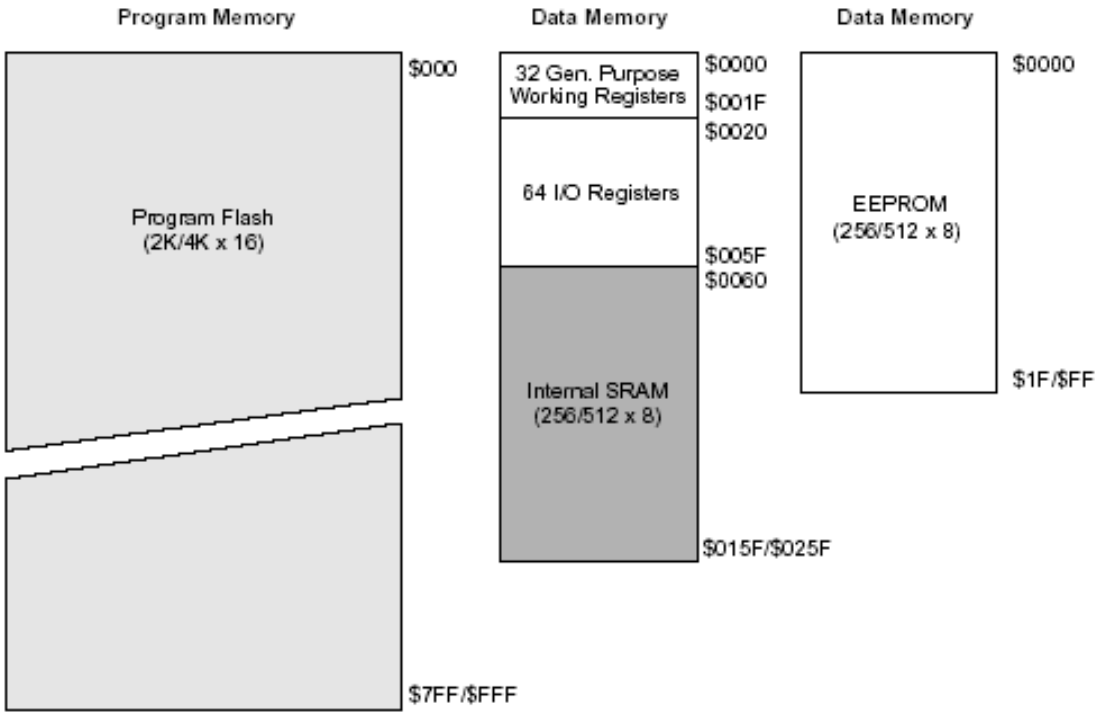
โดยรีจิสเตอร์ R26 – R31 เป็นรีจิสเตอร์ขนาด 8 บิท จำนวน 6 ตัว สามารถจับคู่เพื่อใช้เป็นรีจิสเตอร์ขนาด X, Y และ Z



ภาพที่ 2.14 สถาปัตยกรรมแบบ RISC ของ AT90S4434/8535

ALU จะสนับสนุนการกระทำทางคณิตศาสตร์และลอจิก ระหว่างรีจิสเตอร์กับ รีจิสเตอร์ หรือระหว่างรีจิสเตอร์กับค่าคงที่ ซึ่งการเรียกใช้รีจิสเตอร์ใช้งานทั่วไป สามารถกระทำได้โดยการอ้างหน่วยความจำภายในที่ตำแหน่ง $00 - $1F จำนวน 32 ตำแหน่ง และใน MCU ได้จัดแบ่งให้มีรีจิสเตอร์ที่ใช้ควบคุมการทำงานของหน่วยอินพุตและเอาท์พุตต่างๆ อีก 64 ตำแหน่ง โดยสามารถเรียกใช้งานได้โดยการอ้างตำแหน่งหน่วยความจำที่ตำแหน่ง $20 - $5F ระบบการทำงานของไมโครคอนโทรลเลอร์ใช้หลักการออกแบบของ HAVARD ด้วยการแยกระบบบัสของ PROGRAM และ DATA ออกจากกัน โดยโปรแกรมจะมีการประมวลผลด้วย SINGLE LEVEL PIPELINING ซึ่งทำให้ CPU สามารถ Fetch และ Execute คำสั่งได้ภายใน 1 คาบเวลา

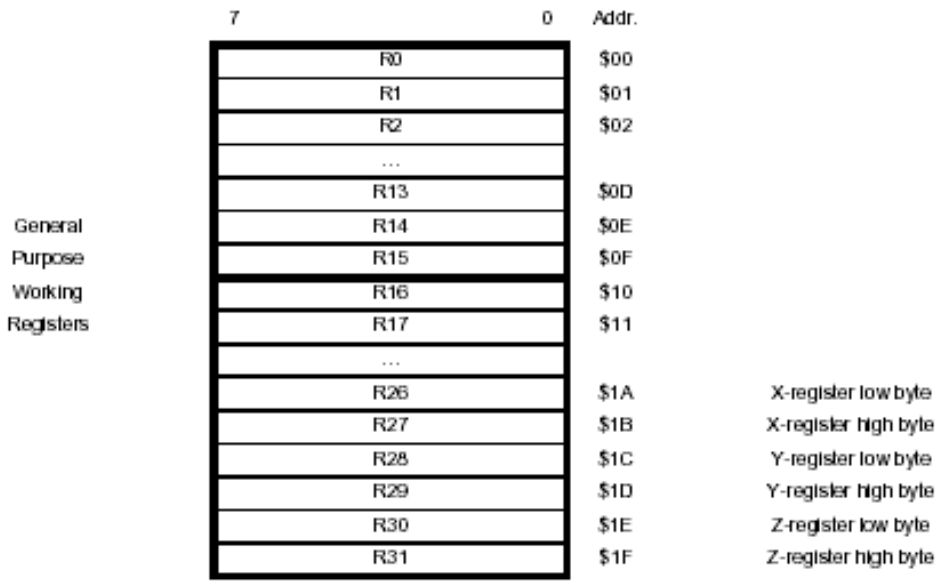
ด้วยคำสั่ง JUMP และ CALL แบบ RELATIVE ที่สามารถกระโดดข้ามการทำงานได้ไกลถึง 2K/4K ซึ่งใน 1 คำสั่งจะใช้รหัสการทำงาน 16 Bit หรือ 1 WORD โดยทุกครั้งที่มีการอินเตอร์รัพท์หรือการข้ามไปทำงานในโปรแกรมย่อยค่าของ PROGRAM COUNTER (PC) จะถูกเก็บลง STACK ซึ่งจะใช้พื้นที่หน่วยความจำใน SRAM บางส่วนเพื่อทำเป็นพื้นที่ของ STACK



ภาพที่ 2.15 โครงสร้างของหน่วยความจำ

รีจิสเตอร์ใช้งานทั่วไป

โครงสร้างรีจิสเตอร์ใช้งานทั่วไปทั้ง 32 ตัว



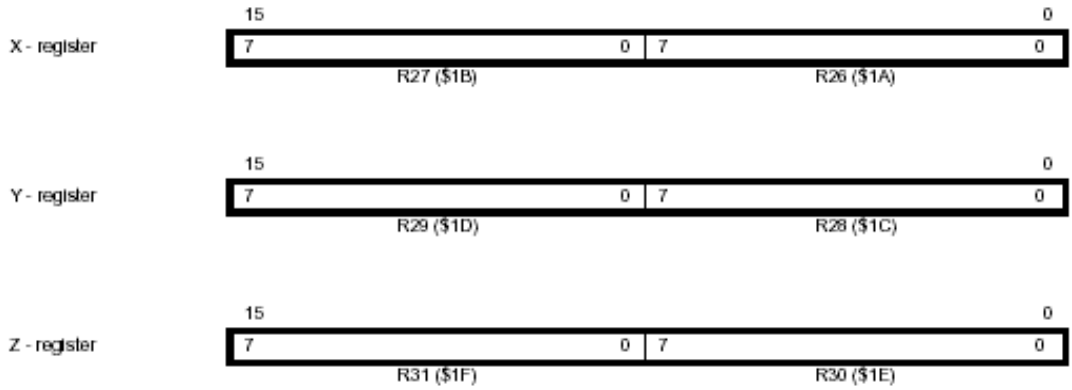
ภาพที่ 2.16 โครงสร้างของรีจิสเตอร์ใช้งานทั่วไป

รีจิสเตอร์ทั้งหมดสามารถใช้ชุดคำสั่ง เพื่อเข้าถึงได้โดยตรงและจะใช้ช่วงเวลาการเข้าถึงเพียง 1 Clock โดยคำสั่ง SBCI, SUBI, CPI, ANDI และ ORI ซึ่งการกระทำระหว่างรีจิสเตอร์กับค่าคงที่และรีจิสเตอร์กับรีจิสเตอร์และคำสั่ง LD1 ที่ใช้โหลดค่าคงที่เข้าในรีจิสเตอร์จะต้องใช้งานกับรีจิสเตอร์ R16 – R31 ส่วนคำสั่ง SBC, SUB, CP, AND และ OR และคำสั่งใช้งานอื่นๆ สามารถใช้งานได้ในรีจิสเตอร์ทั่วไป

ภาพที่ 2.16 เป็นการจัดวางตำแหน่งของรีจิสเตอร์ใช้งานทั่วไปทั้งหมด โดยมีรีจิสเตอร์ที่สามารถนำมาใช้งานเป็นรีจิสเตอร์คู่เพื่อทำเป็นตัวชี้ข้อมูลที่อยู่ในหน่วยความจำ ซึ่งรีจิสเตอร์ในกลุ่มนี้จะใช้ชื่อว่า X, Y และ Z

รีจิสเตอร์ X, รีจิสเตอร์ Y และรีจิสเตอร์ Z

รีจิสเตอร์ R26..R31 สามารถนำมาต่อกันเพื่อทำเป็นรีจิสเตอร์คู่ เพื่อใช้งานเป็นตัวชี้ข้อมูล ในชื่อของรีจิสเตอร์ X, Y และ Z



ภาพที่ 2.17 รีจิสเตอร์ X, Y และ Z

หน่วยประมวลผลทางคณิตศาสตร์และลอจิก

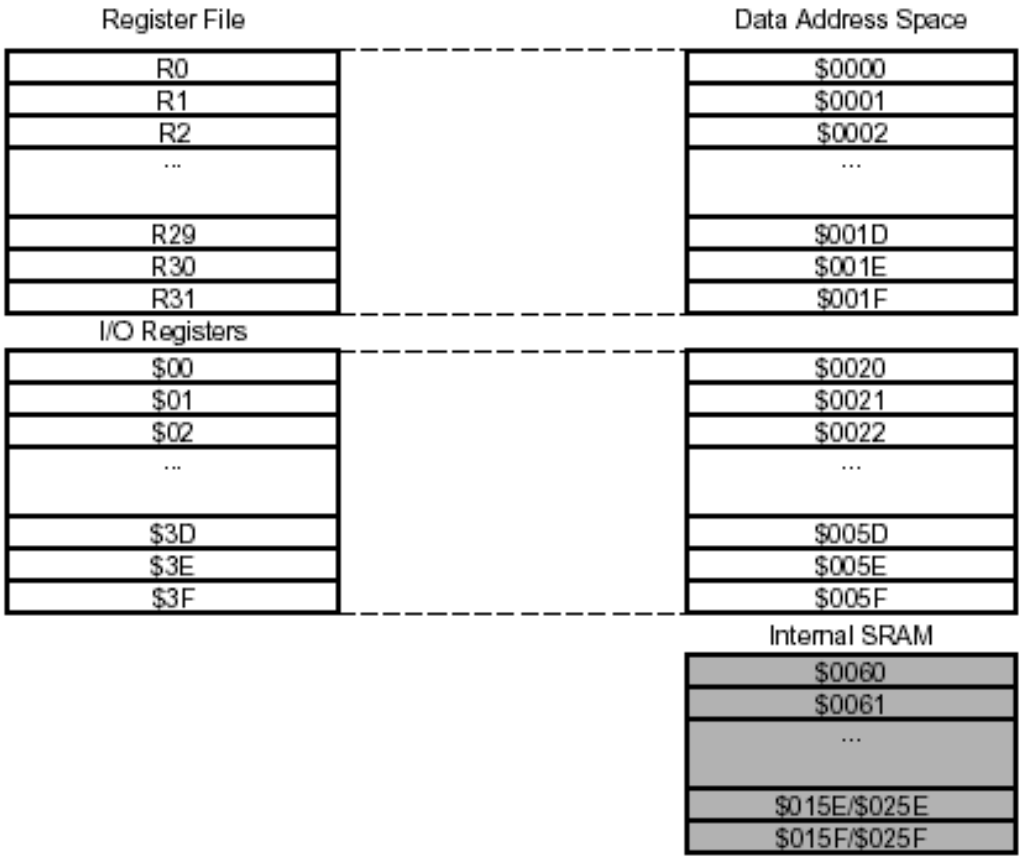
ระบบการประมวลผลที่มีประสิทธิภาพของ AVR คือ ALU สามารถสื่อสารข้อมูลโดยตรงกับรีจิสเตอร์ใช้งานทั่วไปได้ทั้ง 32 ตัว โดย ALU ได้จัดแบ่งระบบการจัดการข้อมูลไว้ 3 ส่วน คือ ส่วนของการจัดการทางคณิตศาสตร์ ส่วนของการกระทำทางลอจิก และในส่วนของการกระทำกับบิท

หน่วยความจำ SRAM

หน่วยความจำภายใน MCU จัดไว้ 608 ตำแหน่ง โดยหน่วยความจำทั้งหมดถูกแบ่งออกเป็นพื้นที่ของรีจิสเตอร์ใช้งานทั่วไป, รีจิสเตอร์ใช้งาน I/O และหน่วยความจำภายใน SRAM

โดย 96 ตำแหน่งแรกจะถูกแบ่งออกเป็นส่วนของรีจิสเตอร์ และอีก 512 ตำแหน่งถูกจัดไว้เป็นส่วนของหน่วยความจำภายใน SRAM

การเข้าถึงข้อมูลถูกแบ่งออกเป็น 5 ส่วน คือ Direct, Indirect with Displacement, Indirect with Pre - Decrement และ Indirect with Post – Increment



ภาพที่ 2.18 การจัดการหน่วยความจำ SRAM

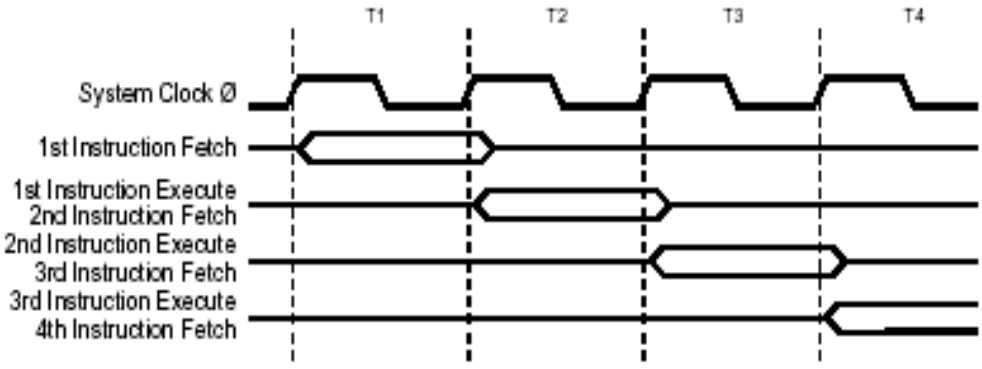
หน่วยความจำข้อมูลแบบ EEPROM

ไมโครคอนโทรลเลอร์ AT90S8535 มีหน่วยความจำแบบ EEPROM ขนาด 512 Byte ซึ่งสามารถอ่านและเขียนครั้งละ 1 Byte โดยสามารถทำการลบและเขียนข้อมูลใหม่ได้ 100,000 ครั้ง

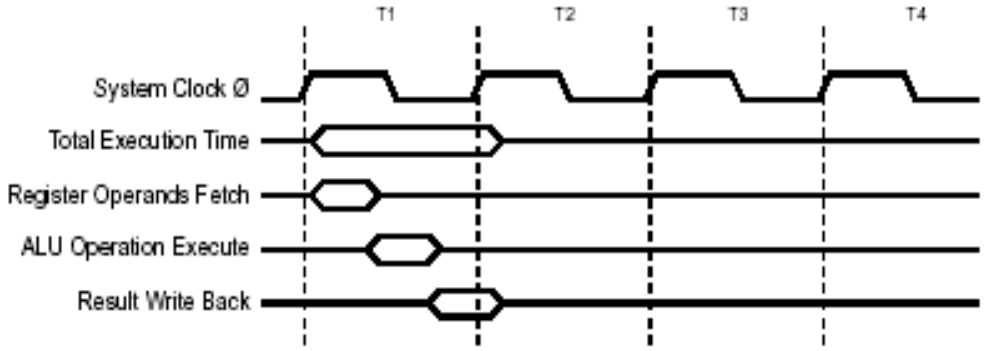
คาบเวลาของการเข้าถึงข้อมูลในหน่วยความจำและการปฏิบัติคำสั่ง

ในหัวข้อนี้ จะอธิบายถึงคาบเวลาของการเข้าถึงข้อมูลในหน่วยความจำและ คาบเวลาของการปฏิบัติคำสั่ง ภาพที่ 2.28 แสดงการ Fetch และ Executions แบบขนานของ CPU ซึ่งทำให้การปฏิบัติคำสั่งได้ 1 MIPS ต่อ MHz

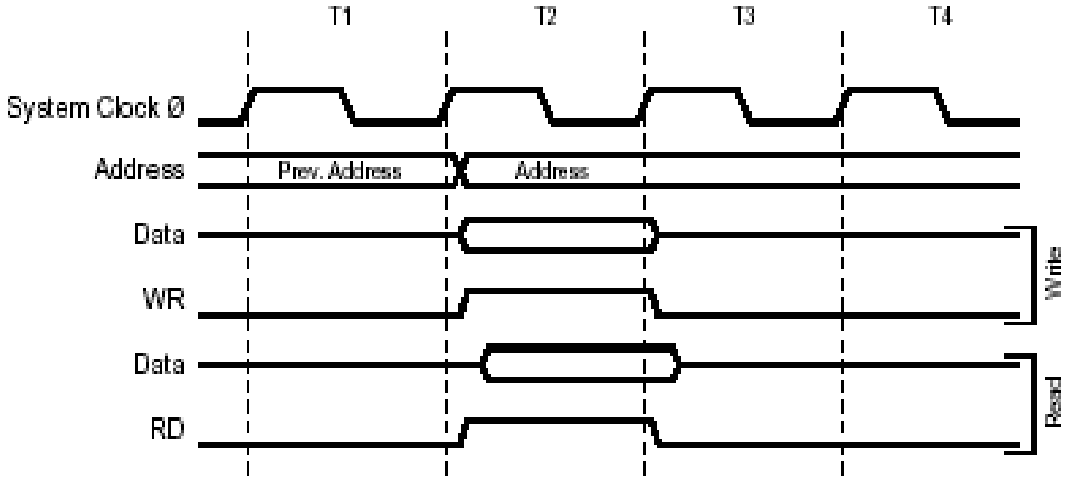
ในภาพที่ 2.20 คาบเวลาการทำงานของ ALU โดยใช้คาบเวลาการปฏิบัติคำสั่งใน 1 Clock การเข้าถึงข้อมูลใน SRAM จะใช้คาบเวลา 2 Clock ซึ่งแสดงใน ภาพที่ 2.21



ภาพที่ 2.19 การ fetch และ Excutions แบบขนาน



ภาพที่ 2.20 การทำงานของ ALU ใน 1 CYCLE



ภาพที่ 2.21 การเข้าถึงข้อมูล SRAM ภายใน CPU

การรีเซ็ตและการอินเตอร์รัพท์

ไมโครคอนโทรลเลอร์ AT90S8353 มีระบบการตอบสนองสัญญาณอินเตอร์รัพท์จาก 16 แหล่งสัญญาณ โดยแยกอินเตอร์รัพท์เวกเตอร์และอินเตอร์รัพท์ออกจากกัน ในการควบคุมการตอบสนองของอินเตอร์รัพท์แต่ละแหล่ง สามารถแยกการควบคุมได้จากบิท Enable

ของอินเตอร์-รัพท์นั้นๆ และบิท 1 ซึ่งใช้ควลคุมอินเตอร์รัพท์ทั้งหมด โดยตำแหน่งแรกๆ ใน PROGRAM MEMORY จะเป็นตำแหน่งที่ถูกใช้เป็นอินเตอร์รัพท์เวกเตอร์และรีเซ็ตซึ่งตารางที่ 2.5 แสดงอินเตอร์รัพท์เวกเตอร์ต่างๆ โดยเริ่มจากอินเตอร์รัพท์ที่มีระดับความสำคัญสูงสุด คือ RESET จนถึงอินเตอร์รัพท์ที่มีระดับความสำคัญต่ำสุด

ตารางที่ 2.5 แสดงอินเตอร์รัพท์เวกเตอร์

Vector No.	Program Address	Source	Interrupt Definition
1	$000	RESET	Hardware Pin and Watchdog Reset
2	$001	INTO	External Interrupt Request 0
3	$002	INT1	External Interrupt Request 1
4	$003	TIMER2 COMP	Timer/Count2 Compare Mach
5	$004	TIMER2 OVF	Timer/Count2 Overflow
6	$005	TIMER1CAPT	Timer/Count1 Capture Event
7	$006	TIMER1 COMPA	Timer/Count1 Compare MachA
8	$007	TIMER1 COMPB	Timer/Count1 Compare MachB
9	$008	TIMER1 OVF	Timer/Count1 Overflow
10	$009	TIMER0 OVF	Timer/Count0 Overflow
11	$00A	SPI, STC	Serial Tranfer Complete
12	$00B	UART, RX	UART, RX Complete
13	$00C	UART, UDRE	UART Data Regiter Emply
14	$00D	UART, TX	UART, TX Complete
15	$00E	ADC	ACD Conversion Complete
16	$00F	EE_RDY	EEPROM Ready
17	$010	ANA_COMP	Analog Comparator

สัญญาณรีเซ็ต

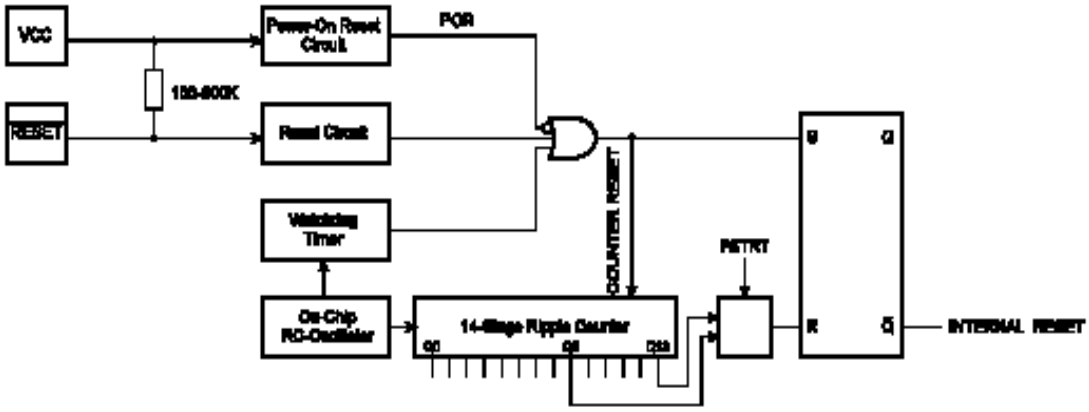
ไมโครคอนโทรลเลอร์ AT90S4434/AT90S8535 มีสัญญาณรีเซ็ต 3 แหล่ง

POWER ON RESET ไมโครคอนโทรลเลอร์จะรีเซ็ตเมื่อมีการจ่ายไฟให้กับขา Vcc และ GND

EXTERNAL RESET ไมโครคอนโทรลเลอร์จะรีเซ็ตเมื่อมีสัญญาณลอจิก LOW เข้ามาที่ขารีเซ็ตเป็นระยะเวลามากว่า 2 คาบเวลาของสัญญาณ XTAL

WATCHDOG RESET ไมโครคอนโทรลเลอร์จะมีรีเซ็ตเมื่อถึงคาบเวลาของ WATCHDOG จะต้องถูกกำหนดให้ทำงาน

ในระหว่างที่เกิดการรีเซ็ต รีจีเตอร์ทั้งหมดจะถูกกำหนดให้ค่าเริ่มต้น และโปรแกรมจะเริ่มทำงานที่ตำแหน่ง $0000 โดยคำสั่งที่ตำแหน่ง $0000 จะต้องเป็นคำสั่ง RJMP แต่ถ้าโปรแกรมไม่มีการกำหนดให้มีการใช้อินเตอร์รัพพ์ พื้นที่ส่วนที่เป็นอินเตอร์รัพพ์เวกเตอร์จะใช้เป็นพื้นที่ของโปรแกรม



ภาพที่ 2.22 โครงสร้างของวงจรรีเซ็ต

ตารางที่ 2.6 คุณลักษณะของสัญญาณรีเซ็ต (5.0 V)

Symbol	Parameter	Min	Typ	Max	Units
Vpot	Power-On Reset Theeshold Voltage	1.4	1.6	1.8	V
Vrst	$\overline{RESET}$ Pin Threshold Voltage		0.6Vcc		V
t_{TOUT}	Reset Delay time Out period FSTRT Unprogrammed	11	16	21	Ms
t_{TOUT}	Reset Delay time Out period FSTRT Unprogrammed	1.0	1.1	1.2	ms

Power On Reset

วงจรของ Power On Reset(POR) ถูกสร้างขึ้นเพื่อให้แน่ใจว่า MCU จะไม่ทำงานถ้าระดับของแรงดันไฟเลี้ยงวงจรยังไม่ถึงระดับที่จะทำให้ระบบการทำงานภายในของ MCU ทำงานได้อย่างถูกต้อง

ถ้าต้องการให้ระบบ Power On Reset ภายใน MCU จะติองต่อขารีเซ็ตเข้ากับ Vcc หรือต่อผ่าน ความต้านทาน Pull up ค่าประมาณ 100k – 500k

โดยบิท FSTRT ใช้ในการเลือกคาบเวลา TIME-OUT ซึ่งเมื่อบิท FSTRT ถูกโปรแกรมจะทำให้ช่วงเวลา TIME-OUT มีคาบเวลาน้อยลง ซึ่งมีประโยชน์อย่างมากเมื่อใช้ OSC ที่เป็น Ceramic Resonator

การสื่อสารอนุกรม

การสื่อสารอนุกรมแบบซิงโครนัส (Serail Peripheral Interfece)

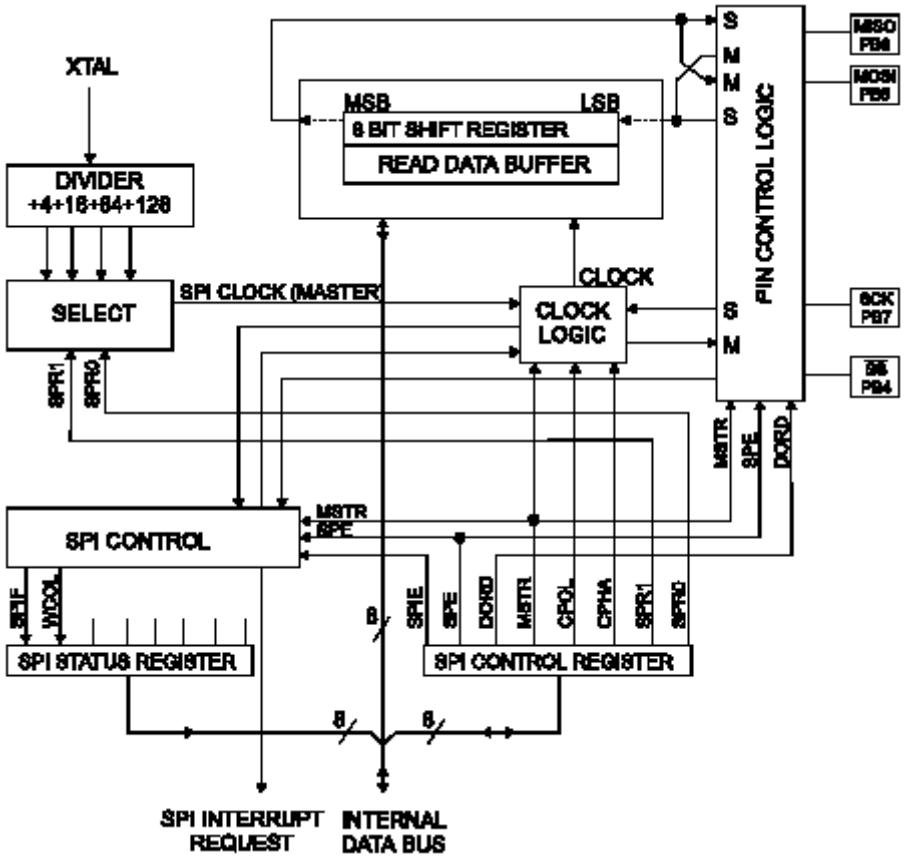
ในไมโครคอนโทรลเลอร์ AT90S4434/AT90S8535 ได้จัดให้มีการสื่อสารแบบ SPI ซึ่งเป็นการสื่อสารแบบโครนัสที่มีความเร็วสูง ซึ่งมีคุณสมบัติดังนี้

เป็นการสื่อสารแบบ FULL DUPLEX ที่ใช้สายสัญญา 3 เส้น ในการส่งถ่ายข้อมูล

สามารถทำงานเป็นได้ทั้งตัวส่งข้อมูลและตัวรับข้อมูล

สามารถเลือกได้ว่าจะให้ส่งบิท MSB หรือ LSB ออกก่อน

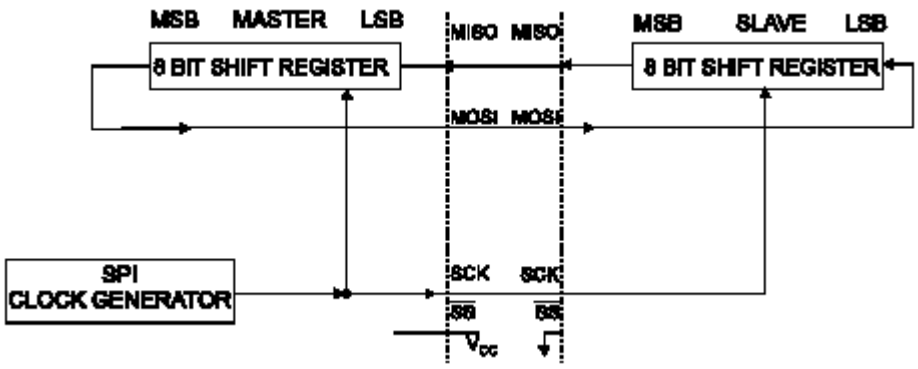
มี FLAG แสดงสถานะการส่งข้อมูลสิ้นสุด



รูปที่ 2.23 แสดงโครงสร้างของ SPI

การสื่อสารข้อมูลระหว่าง MUC ที่เป็น MASTER กับ MCU ที่เป็น SLAVE ด้วยการสื่อสารแบบ SPI โดยขา PB7(SCK) ซึ่งเป็นขา CLOCK จะถูกใช้งานเป็นขาเอาต์พุตเมื่อ MCU เป็น MASTER และถูกใช้งานเป็นขาอินพุทเมื่อ MCU เป็น SLAVE โดยทุกครั้งที่เขียนข้อมูลลงใน SPI Data Register ของ MCU ที่เป็น MASTER และกำเนิด CLOCK ที่ขา SCK และข้อมูลจะเลื่อนออกจากขา MOSI ของตัว MCU ที่เป็น MASTER เข้าไปสู่ขา MOSI ของตัว ที่เป็น SLAVE และเมื่อมีการส่งข้อมูลครบ 1 BYTE สัญญาณ CLOCK จะหยุดและบิท SPIE ในรีสเตอร์ SPCE ถูกเซ็ตเป็น 1 จะทำให้เกิดอินเตอร์รัพท์

ถ้าขา PB4(SS) มีสถานะเป็น LOW จะเป็นการกำหนดให้ MCU ทำหน้าที่เป็น SLAVE โดยเมื่อพิจารณาถึง Shift Register ที่ทำหน้าที่เลื่อนข้อมูลใน MCU ที่ทำหน้าที่เป็น MATER และ SLAVE สามารถที่จะพิจารณาให้เป็น Circular shift Register ขนาด 116 บิท ซึ่งในภาพที่ 2.23 เมื่อต้องการส่งข้อมูลออกจาก MCU ที่เป็น MATER เข้าไปใน MCU ที่เป็น SLAVE ข้อมูลจะถูกส่งถ่ายระหว่าง MPU ทั้ง 2 ตัวในทิศทางที่ตรงข้ามกัน ซึ่ง 1 รอบของการส่งข้อมูลจะทำให้ข้อมูลใน MATER และ SLAVE ถูกแลกเปลี่ยนกัน



ภาพที่ 2.24 Mater-Slave Interconnection

ในด้านส่งข้อมูลจะมี BUFFER เพียงชุดเดียวแต่ในส่วนของด้านรับจะมี BUFFER 2 ชุด เพราะฉะนั้นการส่งข้อมูลจะต้องรอให้ข้อมูลใน Buffer ถูกส่งไปยังด้านรับเรียบร้อยแล้ว จะสามารถเขียนข้อมูลชุดใหม่ลงใน Buffer ได้ ส่วนในการรับข้อมูลจะต้องนำข้อมูลออกจาก SPI Data Register ก่อนที่ข้อมูลชุดใหม่ จะถูกส่งเข้ามาเพื่อไม่ให้ข้อมูล สูญหายสามารถกำหนดทิศทางของ MOSI, MISO, SCK และ SS จะถูกกำหนดตาม ตารางที่ 2.7

ตารางที่ 2.7 SPI Pin Overrides

Pin	Direction, Master SPI	Direction, Slave SPI

MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

SS Pin Function

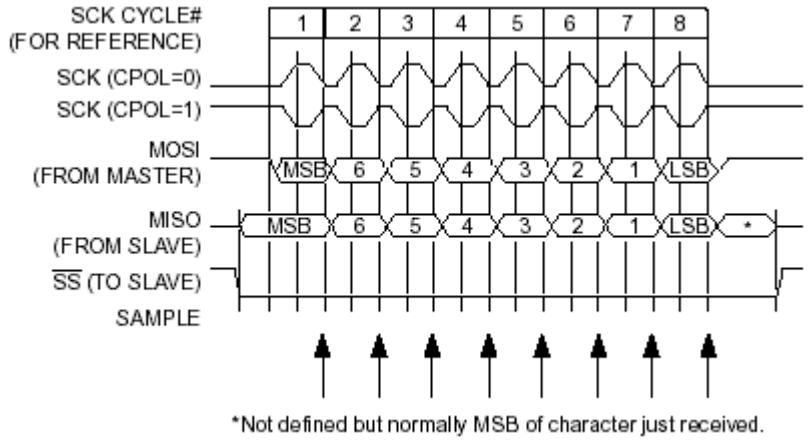
เมื่อ MCU ถูกกำหนดให้เป็น MATER (โดยการเซ็ตบิท MSTR ในรีจีส-เตอร์ SPCR) ซึ่งถ้าขา SS ถูกกำหนดให้เป็นเอาต์พุท (ซึ่งปรกติจะเป็นเอาท์พุท) จะทำให้ MCU เป็น MATER และพร้อมที่จะส่งข้อมูลออก แต่ถ้าขา SS ถูกกำหนดให้เป็นขาอินพุต จะต้องป้อนลอจิก HIGH ให้กับขา SS เพื่อให้แน่ใจว่า MCU จะทำหน้าที่เป็น MATER และพร้อมที่จะส่งข้อมูล แต่ถ้าขา SS ถูกกำหนดเป็นอินพุตและมีสถานะเป็นลอจิก LOW จะทำให้ MCU ถูกควบคุมให้เป็น SLAVE ซึ่งจะทำให้ระบบ SPI ถูกเปลี่ยนแปลงดังนี้

1. บิท MSTR ในรีจีสเตอร์ SPCR จะถูกเคลียร์และ MCU จะกระทำตัวเป็น SLAVE ซึ่งทำให้ขา MOSI และ SCK เป็น อินพุต
2. บิท SPIF ในรีจีสเตอร์ SPSR ถูกเซ็ตและถ้าอินเตอร์รัพท์ของ SPI ได้ถูกกำหนดให้ Enable จะทำให้เกิดการอินเตอร์รัพท์เกิดขึ้น

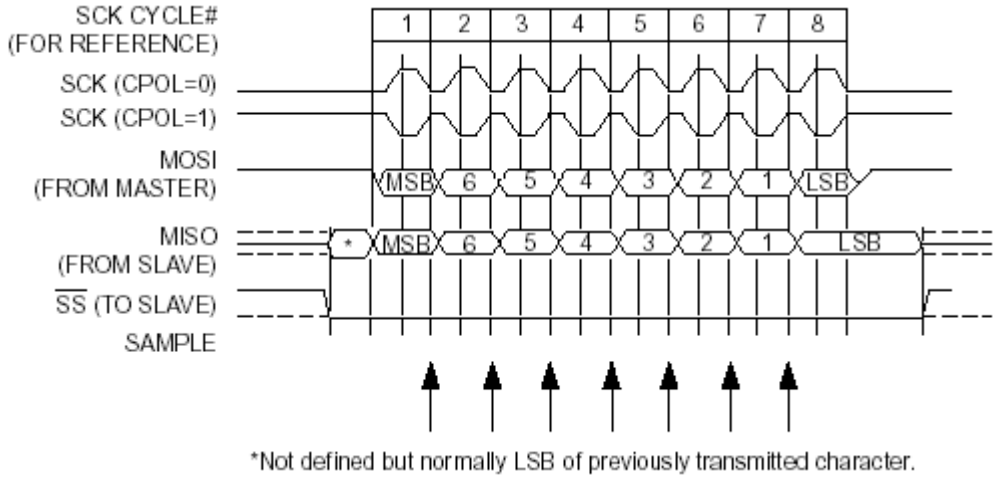
เมื่อ MCU ถูกกำหนดให้เป็น SLAVE (ขา SS เป็นอินพุต) และถูกป้อนลอจิก LOW จะทำให้ MCU พร้อมที่จะรับข้อมูล แต่ถ้าขา SS ถูกป้อนลอจิก HIGH จะทำให้ MCU ไม่สามารถรับข้อมูลได้

DATA MODES

มีวิธีการ 4 อย่างที่จะกำหนดลักษณะสัญญาณที่ขา SCK เพื่อใช้ในการ สื่อสารข้อมูลแบบ SPI โดยสามารถกำหนดได้จากบิท CPHA และบิท CPOL ซึ่งแสดงได้ในภาพที่ 2.24 และ 2.25



ภาพที่ 2.25 SPI Tranfer Format with CPHA=0



ภาพที่ 2.26 SPI Tranfer Format CPHA=1

THE SPI CONTROL REGISTER-SPCR

Bit	7	6	5	4	3	2	1	0	
$0D ($2D)	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
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	

Bit 7-SPIE : SPI Interrupt Enable

เป็นบิทที่ใช้ Enable อินเตอร์รัพท์ของ SPI ถ้าบิท SPIE ถูกเซ็ตเป็น 1 จะเป็นการ Enable อินเตอร์รัพท์ของ SPI ซึ่งบิทนี้จะอยู่ในรีจีสเตอร์ SPIE

Bit 6-SPE:SPI Enable

เป็นบิทที่ใช้ Enable ระบบ SPI โดยถ้าบิทนี้ถูกเซ็ตเป็น 1 จะเป็นการกำหนดให้ระบบ SPI เริ่มทำงาน

Bit 4- DORD: Data Order

บิท DORD เป็นบิทที่ให้ในการควบคุมให้บิท LSB หรือบิท MSB ถูกส่งออกก่อน โดยถ้าบิทนี้ถูกเซ็ตเป็น 1 จะเป็นการกำหนดให้ส่งบิท LSB ออกไปก่อน แต่ถ้าบิทนี้เป็น 0 จะเป็นการกำหนดให้ส่งบิท MSB ออกไปก่อน

Bit 4- MSTR: Mater/Slave select

ถ้าบิท MSTR ถูกเซ็ตเป็น 1 จะเป็นการกำหนดให้ SPI ทำงานเป็น MASTER แต่ถ้า ถูกเคลียร์เป็น 0 จะเป็นการกำหนดให้ SPI ทำงานเป็น SLAVE แต่ถ้าขา SS ถูกกำหนดให้เป็นอินพุตและถูกป้อนลอจิก LOW ใน ขณะที่บิท MSTER เซ็ตเป็น 1 จะทำให้บิท MSTR ถูกเคลียร์ให้เป็น 0 และบิท SPIE ในรีจีสเตอร์ SPSR ถูกเซ็ตเป็น 1 โดย ผู้ใช้สามารถควบคุมให้ SPI เป็น MASTER ได้อีกครั้งโดยการเซ็ตบิท MSTR

Bit 3-CPOL: Clock Polarity

เมื่อบิทนี้เซ็ตเป็น 1 จะมีผลให้ขา SCK มีลอจิกเป็น HIGH เมื่อถูกปล่อยลอย แต่ถ้าบิท CPOL ถูกเคลียร์เป็น 0 ขา จะทำให้ขา SCK มีลอจิก LOW เมื่อถูกปล่อยลอย โดยแสดงไว้ในภาพที่ 29 และ 30

Bit 2-CPHA:Clock Phase

เป็นบิทที่ใช้ควบคุมการกำหนดฟังชั่น แสดงในภาพที่ 2.24 และ 2.25

Bit1,0-SPR1,SPR0:SPI Clock Rate Select 1 and 0

เป็นบิทที่ใช้ในการควบคุมอัตตราของ CLOCK ที่ขา SCK ในขณะที่ SPI เป็น MASTER โดยบิท SPR1 และบิท SPR0 จะไม่มีผลต่อระบบของ SPI กระทำตัวเป็น SLAVE โดยการกำหนดอัตรา CLOCK สามารถกำหนดได้ในตารางที่ 2.8

ตารางที่ 2.8 Relationship Between and the Oscillator

SPR1	SPR0	SCK Frequency
0	0	f_{cl} / 4
0	1	f_{cl} / 16
1	0	f_{cl} / 64
1	1	f_{cl} / 128

THE SPI Stagister-SPSR

Bit	7	6	5	4	3	2	1	0	
$0E ($2E)	SPIF	WCOL	-	-	-	-	-	-	SPSR
Read/Write	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

Bit 7-SPIF: SPI Interrupt Flag

เมื่อข้อมูลถูกส่งเรียบร้อยแล้วบิท SPIF จะถูกเซ็ตให้เป็น 1 และเกิดอินเตอร์รัพท์ ถ้ามีการเซ็ตบิท SPIE ในรีจีสเตอร์ SPCR และบิท I หรือถ้าขา SS ถูกกำหนดให้เป็น อินพุตและถูกป้อนลอจิก LOW ในขณะที่ SPI เป็น MASTER จะทำให้บิท SPIF ถูกเซ็ต และจะถูกเคลียร์โดยอัตโนมัติเมื่อ CPU ทำโปรแกรมบริการ อินเตอร์รัพท์

Bit 6-WCOL:Write Collision flag

บิทนี้จะถูกเซ็ตเป็น 1 ถ้ารีจีสเตอร์ SPDR ถูกเขียนในขณะที่มีการส่งข้อมูล

Bit 5..O:Reserved bits

บิทนี้จะถูกสงวนไว้ใน AT90s4434/AT90S8535 โดยปรกติจะเป็น 0

The SPI Data Register- SPDR

Bit	7	6	5	4	3	2	1	0	
$0F ($2F)	MSB							LSB	SPDR
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	

เป็นรีจีสเตอร์ที่สามารถอ่าน และเขียนข้อมูลได้ใช้ในการเก็บข้อมูลระหว่างการสื่อสารข้อมูลแบบSPI

การสื่อสารข้อมูลแบบอซิงโครนัส (UART)

ไมโครคอนโทรลเลอร์ AT90S4434/8535 ถูกจัดให้มีฟังก์ชั่นในส่วนของการสื่อสารแบบอะซิงโครนัส หรือ UART (Asynchronous Receiver and Transmitter) โดยได้จัดให้มีคุณสมบัติหลักๆ ดังนี้

สามารถเปลี่ยนแปลง BAUD RATE ของการสื่อสารได้หลาย BAUD RATE

สามารถสื่อสารได้ในอัตรา BAUD RATE ที่สูงในขณะที่ความถี่ XTAL ต่ำ

สื่อสารข้อมูลได้ทั้ง 8 บิท และ 9 บิท

มีส่วนของการกำจัดสัญญาณรบกวน (Noise filltering) ที่เกิดขึ้นในสายส่ง

ตรวจจับความผิดของการสื่อสารข้อมูล ที่เกิดจากการที่ข้อมูลที่ถูกส่งเข้ามาก่อน ยังไม่ได้ถูกอ่านออกจากบัฟเฟอร์และมีข้อมูลใหม่เข้ามาทับ (Overrun detection)

ตรวจจับความผิดพลาดของกรอบข้อมูลผิดพลาด (Framimg error detection)

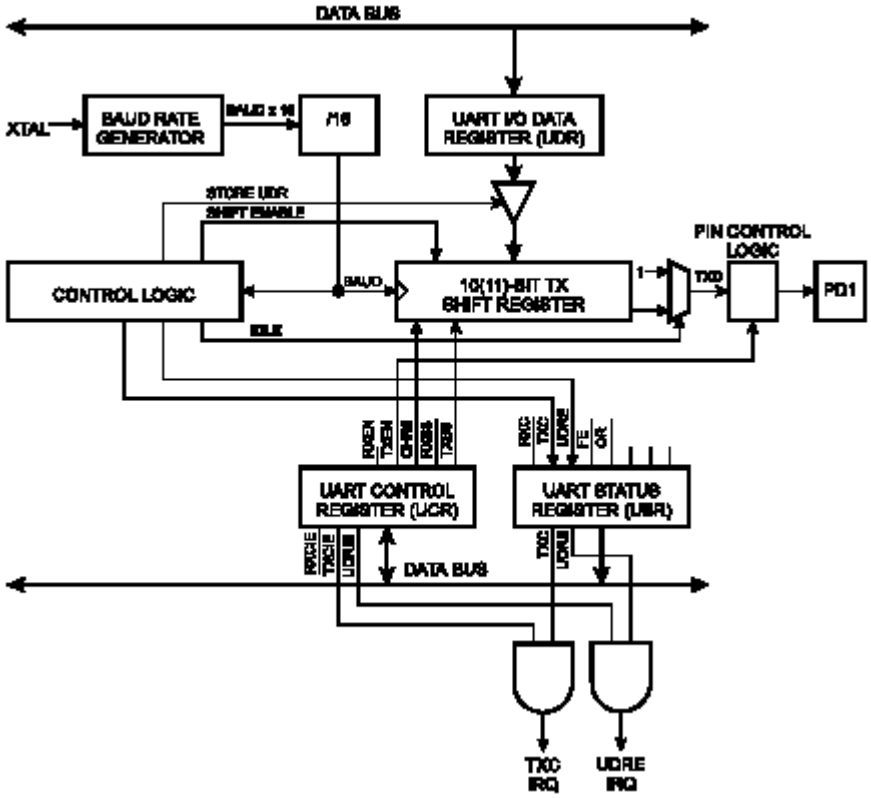
ตรวจจับความผิดพลาดของบิทเริ่มต้น (False Start Bit detection)

จัดให้มีการแยกระบบอินเตอร์รัพท์ของการสื่อสารออกเป็น 3 ส่วน คือ อินเตอร์รัพท์ที่เกิดจากการส่งข้อมูลอินเตอร์รัพท์ที่เกิดจากการรับข้อมูล และอินเตอร์รัพท์ที่เกิดจากการไม่มีข้อมูลในรีจีสเตอร์

มีบัฟเฟอร์ในการเก็บข้อมูลที่จะออกและข้อมูลที่รับ

การส่งข้อมูล (Data Tranmission)

BLOCK DIAGRAM การส่งข้อมูลแสดงในภาพที่ 2.27



ภาพที่ 2.27 ไดอแกรมการส่งข้อมูลแบบ UART

การเริ่มต้นส่งข้อมูล สามารถกระทำได้ โดยการเขียนข้อมูลลงในรีจีสเตอร์ UDR (UART I/O Data Register) โดยข้อมูลจะถูกจะถูกถ่ายโอนจากรีสเตอร์ UDR เข้าไปยัง TRANSMIT SHIFT REGISTER เพื่อเลื่อนข้อมูลออกไปยังสายสัญญาณ ซึ่งการถ่ายข้อมูลจาก รีจีสเตอร์ UDR เข้าไปยัง TRANSMIT SHIFT REGISTER มีอยู่ 2 กรณี คือ

ถ้าข้อมูลที่ถูกเขียนลงในรีจีสเตอร์ UDR ถูกเขียนภายหลังจากที่บิท STOP ของข้อมูลที่ถูกส่งก่อนหน้านี้ถูกส่งออกไป ซึ่งจะทำให้ข้อมูลที่อยู่ในรีจีสเตอร์ UDR ถูกโหลดลงใน TRANSMIT SHIFT REGISTER โดยทันที ที่ข้อมูลนั้นถูกเขียนลงในรีจีสเตอร์ UDR

ถ้าข้อมูลที่ถูกเขียนลงในรีจีสเตอร์ UDR ถูกเขียนก่อนหน้าที่บิท STOP ของข้อมูลที่ถูกส่งก่อนหน้านี้จะถูกส่งออกไป ซึ่งจะทำให้ข้อมูลที่อยู่ใน UDR ถูกโหลดลง TRANSMIT SHIFT REGISTER ในช่วงเวลาที่บิท STOP ขอลข้อมูลก่อนหน้านี้กำลังถูกส่งออกไป

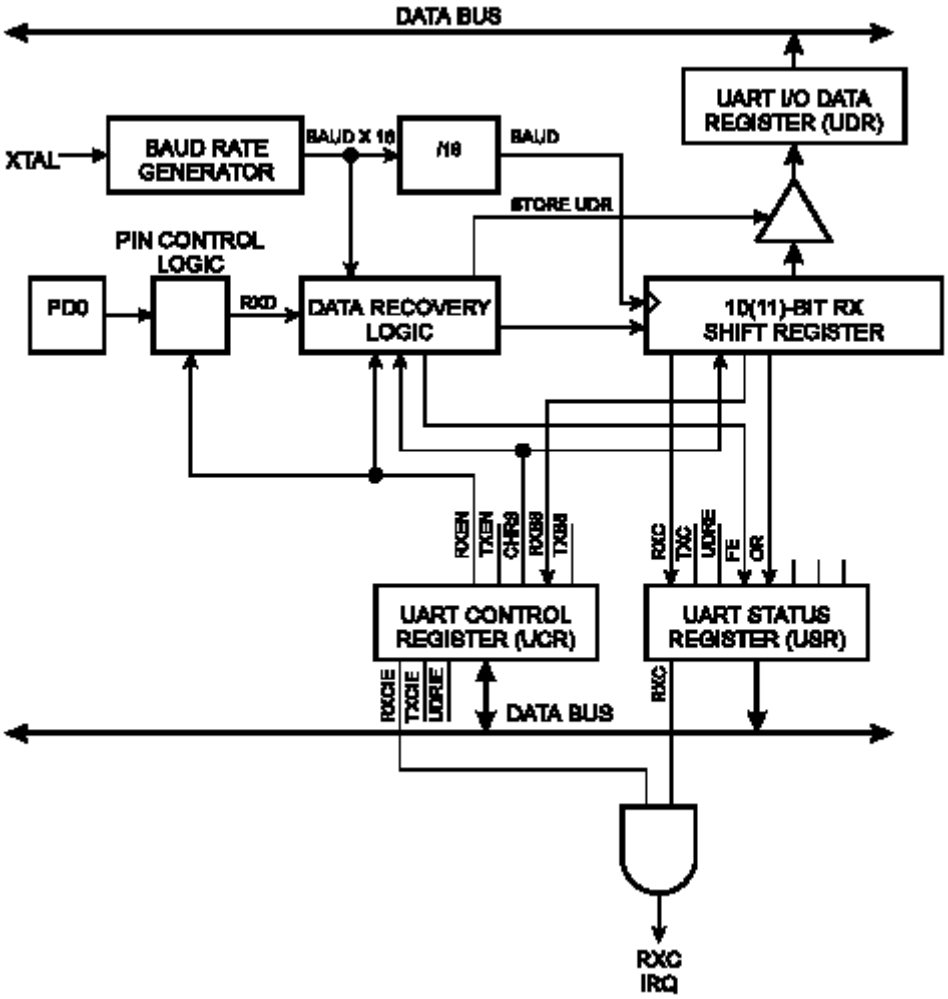
ถ้าใน TRANSMIT SHIFT REGISTER ไม่มีข้อมูลจะทำให้บิท UDRE (UART Data Register Empty) ในรีจีสเตอร์ USR เพื่อส่งออก โดยในขั้นตอนต่อไปถ้าข้อมูล ถูกลดลงใน TRANSMIT SHIFT REGISTER จะทำให้ บิทแรกที่จะถูกส่งซึ่งเป็นบิท START จะถูกเคลียร์ โดยบิทที่ 9

(ถ้าเป็นการส่งข้อมูลขนาด 10 บิท) หรือบิทที่ 10 (ถ้าเป็นการส่งข้อมูลขนาด 11 บิท) ซึ่งเป็น บิท STOP จะถูกเซ็ตให้เป็น 1

ในเวลาของการเริ่มต้นส่งข้อมูลจะทำให้บิท START ถูกส่งออกที่ ขา TXD และตามด้วยข้อมูลที่เป็น LSB บิท โดยเมื่อบิท STOP ถูกส่งออกจะทำให้ TRANSMIT SHIFT REGISTER โหลดข้อมูลชุดใหม่จากรีจีสเตอร์ UDR แต่ถ้าภายหลังจากที่มีการส่งบิท STOP ออกไปแล้วและไม่มีข้อมูลใหม่ที่จะทำการส่งจะทำให้บิท TX Complete Flag และ TXC ในรีจีสเตอร์ USR ถูกเซ็ต

ถ้าบิท TXEN ถูกเซ็ต จะเป็นการกำหนดให้ขา TX ทำหน้าที่ในการ สื่อสารข้อมูลแบบอนุกรม แต่ถ้าบิท TXEN ถูกเคลียร์จะทำให้ขา TX ถูกกำหนดให้เป็น ขา I/O

การรับข้อมูล DATA RECETION



ภาพที่ 2.28 แสดงไดอะแกรมการรับข้อมูลของ UART

ภาพที่ 2.28 แสดง BLOCK DIAGRAM ของการรับข้อมูล ภาคหน้าของการรับข้อมูลจะเป็นส่วนที่ทำหน้าที่สุ่มข้อมูลที่เข้ามาที่ขา RXD ด้วยอัตรา 16 เท่าของ BAUD RATE เมื่อสายสัญญาณถูกปล่อยให้ว่าง ส่วนรับข้อมูลจะสุ่มหาขอบขาลงของบิท START โดยสัญญาณที่สุ่มพบการเปลี่ยนแปลงจากลอจิก 1 เป็น ลอจิก 0 (ขอบขาลง) ของสัญญาณในสายส่งก็จะถูกกำหนดให้เป็นการสุ่มครั้งแรก โดยสัญญาณการสุ่มที่ 8, 9 และ 10 จะถูกนำมาเป็นตัวตัดสินว่าระดับสัญญาณที่ทำการสุ่มเป็น ลอจิกใด ซึ่งถ้าเป็นการตรวจสอบหาบิท START และ สัญญาณที่ได้จากการสุ่ม ครั้งที่ 8,9 และ 10 โดย 2 ใน 3 ครั้งได้ ลอจิก 1 จะทำระบบรับข้อมูลเริ่มหาบิท START ใหม่ โดยถือว่าบิทที่สุ่มพบเป็นสัญญาณรบกวน แต่ถ้าบิท STOP ถูกส่งเข้ามาและสัญญาณที่ได้จากการสุ่ม 2 ใน 3 ครั้ง ไม่ได้สัญญาณลอจิก 1 จะทำให้ FRAMING

ERROR (FE) บิท ในรีจีสเตอร์ USR (UART Status Register) ถูกเซ็ต โดยก่อนที่ผู้ใช้จะทำการอ่านข้อมูลจากรีจีสเตอร์ UDR ผู้ใช้จะต้องทำการตรวจสอบบิท FE ก่อน

ข้อมูลที่รับได้จากสายส่งจะถูกส่งไปยังรีจีสเตอร์ UDR และบิท RXC ในรีจีสเตอร์ USR จะเซ็ต ซึ่งรีจีสเตอร์ UDR จะถูกแบ่งออกเป็น 2 ส่วน โดยส่วนหนึ่งจะใช้ในการส่งข้อมูล และอีกส่วนหนึ่งจะใช้ในการรับข้อมูล ถ้าภายหลังจากที่ทำการอ่านรับข้อมูลเรียบร้อยแล้วและยังไม่ได้ทำการอ่านข้อมูลออกจากรีจีสเตอร์ UDR และบิท OVER RUN (OR) ใน รีจีสเตอร์ UCR ถูกเซ็ต ซึ่งแสดงว่าข้อมูล BYTE สุดท้ายที่อยู่ในรีจีสเตอร์ UDR เกิดการ สูญหาย

ถ้าบิท RXEN ในรีจีสเตอร์ UCK ถูกเซ็ตเป็น 1 จะเป็นการกำหนดให้ขา RXD ทำหน้าที่เป็นขาในการรับข้อมูลแบบอนุกรม แต่ถ้า บิท RXEN ถูกเคลียร์ให้เป็น 0 จะทำให้ขา RXD ทำหน้าที่เป็น ขา I/O

การควบคุม UART

The UART I/O Data Register-UDR

Bit	7	6	5	4	3	2	1	0	
$0C ($2C)	MSB							LSB	UDR
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	

โครงสร้างจะถูกแยกออกเป็น 2 ส่วนแต่ใช้ตำแน่งที่อยู่เดียวกัน ถ้าเป็นการเขียนข้อมูลลงในรีจีสเตอร์จะเป็นรีจีสเตอร์ที่ใช้ในการเก็บข้อมูลจากรีจีสเตอร์ จะเป็นการอ่านข้อมูลที่รับได้จาก UART

The UART Status Register-USR

Bit	7	6	5	4	3	2	1	0	
$0B ($2B)	RXC	TXC	UDRE	FE	OR	-	-	-	USR
Read/Write	R	R/W	R	R	R	R	R	R	
Initial value	0	0	1	0	0	0	0	0	

เป็นรีจีสเตอร์ที่สามารถอ่านได้อย่างเดียว โดยข้อมูลภายในจะแสดงสถานะต่างๆ ของ UART

Bit 7-RXC:UART Receive Complete

บิทนี้จะเซ็ตเป็น 1 เมื่อมีการรับข้อมูลจากพอร์ตอนุกรมแล่ะข้อมูลเก็บลงในรีจีสเตอร์ UDR เสร็จเรียบร้อย ซึ่งบิท RXC จะเซ็ตโดยไม่คำนึงถึงการตรวจสอบความผิดพลาดของกรอบข้อมูลว่าเกิดความผิดพลาดหรือไม่โดยเมื่อบิท RXCIE ในรีจีสเตอร์ UCR และบิท RXC เซ็ตเป็น 1 จะทำให้เกิดสัญญาณอินเตอร์รัพท์เกิดขึ้น โดยบิท RXC จะถูกเคลียร์ให้เป็น 0 เมื่อค่าในรีจีสเตอร์ UDR ถูกอ่าน

Bit 6- TXC: UART Transmit Complete

บิทนี้ถูกเซ็ตเป็น 1 เมื่อข้อมูลถูกส่งออก(รวมบิท STOP) ทางพอร์ตทั้งหมดและไม่มีข้อมูลชุดใหม่เขียนลงในรีจีสเตอร์ UDR ซึ่งสถานะของบิทนี้จะมีประโยชน์มากในการส่งข้อมูลอนุกรมแบบ HALF-DUPLEX

เมื่อบิท TXCIE ในรีจีสเตอร์ UCK และบิท TXC เซ็ตเป็น 1 จะทำให้เกิดอินเตอร์รัพท์ ซึ่งบิท TXC สามารถเคลียร์ให้เป็น 0 ได้โดยการเขียนลอจิก 1 ลงในบิท TXC หรือเคลียร์โดยอัตโนมัติ เมื่อ CPU ทำโปรแกรมบริการอินเตอร์รัพท์

Bit5-UDRE:UART Data Register Empty

บิทนี้จะถูกเซ็ตเป็น 1 เมื่อเข้ามูลจาก Data Register ถูกเขียนลงใน shift Register ซึ่งหมายถึงไม่มีข้อมูลใน Data Register เพื่อแสดงให้ผู้ใช้เขียนข้อมูลใหม่ลงใน Data Register เพื่อส่งข้อมูลชุดต่อไป เมื่อบิท UDRIE และบิท UDRE เซ็ตเป็น 1 จะทำให้เกิดสัญญาณอินเตอร์รัพท์ โดยบิท UDRE จะเคลียร์เป็น 0 เมื่อเขียนข้อมูลลงในรีจีสเตอร์ UDR

Bit 4-FE Framig Error

บิทนี้จะถูกเซ็ตเป็น 1 เมื่อมีการตรวจพบว่ากรอบข้อมูลผิดพลาด

Bit 3-OR: Overrun

บิทนี้จะถูกเซ็ตเมื่อตรวจสอบได้ว่าเกิด Overrun คือ การที่ข้อมูลชุดใหม่เข้ามาทับข้อมูลเดิม

Bit 2..0-Res: Reserved bits

บิท 0 – บิท 2 ถูกสงวนไว้ โดยปกติจะเป็น 0

The UART Conteol Register-UCR

Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8	UCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	W	
Initial value	0	0	0	0	0	0	0	0	

Bit 7-RXCIE:RX Complete Interrupt Enble

เป็นบิทที่ใช้ Enble อินเตอร์รัพท์ ของการรับข้อมูล คือ เมื่อการรับข้อมูลไม่มีความผิดพลาดจะทำให้บิท RXC เซ็ตเป็น 1 ซึ่งจะทำให้เกิดอินเตอร์รัพท์ถ้าบิท RXCIE และบิท 1 เซ็ตเป็น 1

Bit 6-TXCIE: TX Complete Interrupt Enable

เป็นบิทที่ใช้ Enable อินเตอร์รัพท์ ของการส่งข้อมูล คือ เมื่อมีการรับ ข้อมูลครบทุกบิทจะทำให้บิท TXC เซ็ตเป็น 1 ซึ่งจะทำให้เกิดอินเตอร์รัพท์ถ้าบิท TXCIE และบิท 1 เซ็ตเป็ น 1

Bit 5-UDRIE : UART Data Register Empty Interrupt Enable

ถ้าไม่มีข้อมูลในรีจีสเตอร์ UDR จะทำให้บิท UDRE เซ็ตเป็น 1 ซึ่งจะทำให้เกิดอินเตอร์รัพท์ถ้าบิท UDRIE และบิท I เซ็ตเป็น 1

Bit 4-RXEN: Receiver Enable

ใช้กำหนดให้ UART เริ่มรับข้อมูล

Bit 3- TXEN: Transmitter Enable

ใช้กำหนดให้ UART เริ่มส่งข้อมูล

Bit 2- CHR9:9 Bit Characters

เป็นบิทที่กำหนดให้มีการรับส่งข้อมูลขนาด 9 บิท ถ้าบิทนี้เซ็ตเป็น 1 จะเป็ฯการกำหนดให้มีการรับส่งข้อมูลขนาด 9 บิท แต่ถ้าบิท CHR9 ถูกเคลียร์เป็น 0 จะเป็นการรับข้อมูลขนาด 8 บิท

Bit 1- RXB8: Recive Data Bit 8

บิท RXB8 จะทำหน้าที่ในการเก็บค่าบิทที่ 9 ของการรับข้อมูลขนาด 9 บิท

Bit 0- TXB8 : Transmit Data Bit 8

บิท TXB 9 จะทำหน้าที่ในการเก็บที่ในการเก็บค่าบิทที่ 9 ของการส่ง ข้อมูล ขนาด 9 บิท

การกำหนด BAUD RATE

$$BAUD \quad \frac{f_{CK}}{16(UBRR \quad 1)} \tag{2.1}$$

BAUD = Baud-Rate

f_{CK} = Crystal Clock frequency

UBRR = Contents of the UART Baud Rate register, UBRR (0.255)

รีจีสเตอร์ UBRR (UART Baud Rate Register)

เป็นรีจีสเตอร์ขนาด 8 บิท ที่ใช้ในการกำหนดอัตตรา Baud Rate

Bit	7	6	5	4	3	2	1	0	
$09 ($29)	MSB							LSB	UBRR
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	

ตารางที่ 2.9 แสดงการกำหนดค่า Baud Rate

Baud Rate	1 MHz	%Error	1.8432 MHz	%Error	2 MHz	%Error	2.4576 MHz	%Error
2400	UBRR= 25	0.2	UBRR= 47	0.0	UBRR= 51	0.2	UBRR= 47	0.0
4800	UBRR= 12	0.2	UBRR= 23	0.0	UBRR= 25	0.2	UBRR= 23	0.0
9600	UBRR= 6	7.5	UBRR= 11	0.0	UBRR= 12	0.2	UBRR= 11	0.0
14400	UBRR= 3	7.8	UBRR= 7	0.0	UBRR= 8	3.7	UBRR= 7	3.1
19200	UBRR= 2	7.8	UBRR= 5	0.0	UBRR= 6	7.5	UBRR= 5	0.0
28800	UBRR= 1	7.8	UBRR= 3	0.0	UBRR= 3	7.8	UBRR= 3	6.3
38400	UBRR= 1	22.9	UBRR= 2	0.0	UBRR= 2	7.8	UBRR= 2	0.0
57600	UBRR= 0	7.8	UBRR= 1	0.0	UBRR= 1	7.8	UBRR= 1	12.5
76800	UBRR= 0	22.9	UBRR= 1	3.3	UBRR= 1	22.9	UBRR= 1	0.0
115200	UBRR= 0	84.3	UBRR= 0	0.0	UBRR= 0	7.8	UBRR= 0	25.0

Baud Rate	3.2768 MHz	%Error	3.6864 MHz	%Error	4 MHz	%Error	4.608 MHz	%Error
2400	UBRR= 84	0.4	UBRR=	0.0	UBRR=	0.2	UBRR=	0.0
4800	UBRR= 42	0.8	95	0.0	103	0.2	119	0.0
9600	UBRR= 20	1.6	UBRR=	0.0	UBRR=	0.2	UBRR=	0.0
14400	UBRR= 13	1.6	47	0.0	51	2.1	59	0.0
19200	UBRR= 10	3.1	UBRR=	0.0	UBRR=	0.2	UBRR=	0.0
28800	UBRR= 6	1.6	23	0.0	25	3.7	29	0.0
38400	UBRR= 4	6.3	UBRR=	0.0	UBRR=	7.5	UBRR=	6.7
57600	UBRR= 3	12.5	15	0.0	16	7.8	19	0.0
76800	UBRR= 2	12.5	UBRR=	0.0	UBRR=	7.8	UBRR=	6.7
115200	UBRR= 1	12.5	11	0.0	12	7.8	14	20.0
			UBRR=		UBRR=		UBRR=	
			7		8		9	
			UBRR=		UBRR=		UBRR=	
			5		6		7	
			UBRR=		UBRR=		UBRR=	
			3		3		4	
			UBRR=		UBRR=		UBRR=	
			2		2		3	
			UBRR=		UBRR=		UBRR=	
			1		1		2	

Baud Rate	7.3728 MHz	%Error	8 MHz	%Error	9.216 MHz	%Error	11.059 MHz	%Error

2400	UBRR=	0.0	UBRR=	0.2	UBRR=	0.0	UBRR=	-
4800	191	0.0	207	0.2	239	0.0	287	0.0
9600	UBRR= 95	0.0	UBRR=	0.2	UBRR=	0.0	UBRR=	0.0
14400	UBRR= 47	0.0	103	0.8	119	0.0	143	0.0
19200	UBRR= 31	0.0	UBRR=	0.2	UBRR=	0.0	UBRR=	0.0
28800	UBRR= 23	0.0	51	2.1	59	0.0	71	0.0
38400	UBRR= 15	0.0	UBRR=	0.2	UBRR=	0.0	UBRR=	0.0
57600	UBRR= 11	0.0	34	3.7	39	0.0	47	0.0
76800	UBRR= 7	0.0	UBRR=	7.5	UBRR=	6.7	UBRR=	0.0
115200	UBRR= 5	0.0	25	7.8	29	0.0	35	0.0
	UBRR= 3		UBRR=		UBRR=		UBRR=	
			16		19		23	
			UBRR=		UBRR=		UBRR=	
			12		14		17	
			UBRR=		UBRR=		UBRR=	
			8		9		11	
			UBRR=		UBRR=		UBRR=	
			6		7		8	
			UBRR=		UBRR=		UBRR=	
			3		4		5	

Maximum Baud rate to each frequency

พอร์ตอินพุตและเอาท์พุต

พอร์ต A

เป็นพอร์ต 2 ทิศทางขนาด 8 บิท โดยมีหน่วยควบคุมการทำงานต่างๆ ของพอร์ต คือ รีจีสเตอร์ PORTA(DATA REGISTER) อยู่ที่ตำแหน่ง $IB($3B), รีจีสเตอร์ DDRA (DATA DIRECTION REGISTER) อยู่ที่ตำแหน่ง $IA($3A) และPINA(PORT A INPUT PINS) อยู่ที่ตำแหน่ง $19 ($39) โดย PINA จะสามารถอ่านได้อย่างเดียว ไม่สามารถเขียนข้อมูลลงได้ ในขณะที่ PORTA และ DDRA สามารถอ่านและเขียนได้ โดยแต่ละขาสัญญาณของพอร์ต A สามารถกำหนดให้ความต้านทาน PULL-UP ทำงานในสภาวะ ACTIVE หรือไม่ ACTIVE ก็ได้โดยแต่ละขาสัญญาณของพอร์ต A จะสามารถรับกระแส (SINK CURRENT) ได้ 20 mA ซึ่งถ้าขาของพอร์ต A ใช้งานเป็นอินพุตและภายนอกมีความต้านทาน PULL LOW ในขณะที่ความต้าน PULL UP ภายใน ACTIVEจะทำให้ MCU จ่ายกระแสออกมาภายนอก

พอร์ต A จะถูกนำไปใช้งานอีกอย่างหนึ่งคือ เป็นขาสัญญาณอินพุทของวงจร ANALOG TO DIGITAL โดยถ้าบางขาของพอร์ต A ถูกใช้งานเป็นขาเอาท์พุตของพอร์ต ในขณะที่บางขาของพอร์ต A ถูกใช้เป็นขาอินพุตของ ANALOG TO DIGITAL ผู้ใช้จะต้องไม่เปลี่ยนแปลงสภาวะการทำงานของพอร์ต A ในขณะที่วงจร ANALOG TO DIGITAL กำลัง Conversion สัญญาณ ซึ่งอาจจะทำให้การทำการแปลงสัญญาณถูกขัดจังหวะ จะมีผลให้การแปลงสัญญาณ ผิดพลาด การทำงานในโหมดประหยัดพลังงานจะทำให้ SCHMITT TRIGGER ของส่วนอินพุต ถูกตัดออก The Port A Data-PORT A

The Port A Data Register - PORTA

Bit	7	6	5	4	3	2	1	0	
$1B ($3B)	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
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	

The Port A Data Direction Register - DDRA

Bit	7	6	5	4	3	2	1	0	
$1A ($3A)	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
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	

The Port A Input Pins Address - PINA

Bit	7	6	5	4	3	2	1	0	
$19 ($39)	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

PINA ไม่ใช่รีจีสเตอร์ ซึ่งตำแหน่งที่กำหนดจะเป็นตำแหน่งของแต่ละขาของพอร์ต โดยเมื่ออ่าน PORTA จะเป็นการอ่านค่าที่ PINA จะเป็นการอ่านค่าจริงๆ ของขาพอร์ต

การใช้งาน พอร์ต A เป็นขาอินพุตเอาท์พุต

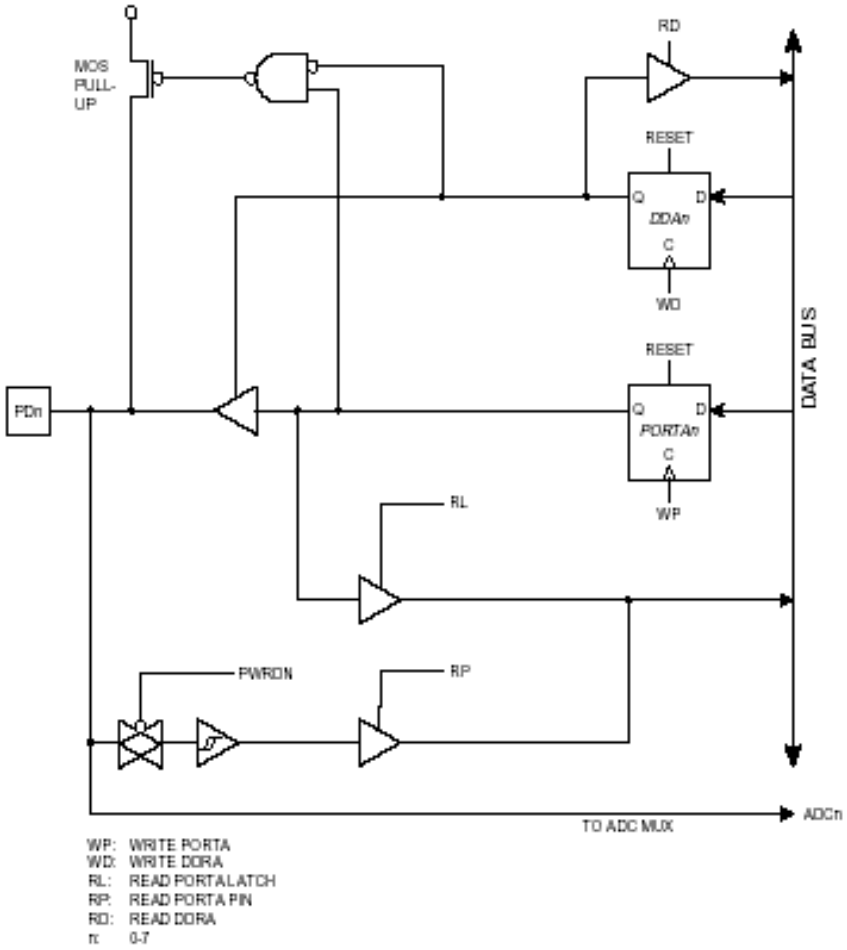
บิท DDAn ในรีจีสเตอร์ DDRA จะเป็นบิทที่ใช้ในการกำหนดทิศทางของแต่ละพอร์ต ถ้าบิท DDAn ถูกเซ็ตเป็น 1 จะทำให้ขาของพอร์ตนั้นเป็นเอาท์พุต แต่เมื่อ DDAn ถูกเคลียเป็น 0 จะทำให้ขานั้นถูกกำหนดเป็นอินพุต ซึ่งสภาวะของขาต่างๆ แสดงดังตารางที่ 2.10

ตารางที่ 2.10 DDAn Effects on PORTA Pins

DDAn	PORTAn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PAn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

โครงสร้างของ พอร์ต A

ทุกขาทั้งหมดของพอร์ต A จะมีลักษณะโครงสร้างดังภาพที่ 2.29



ภาพที่ 2.29 โครงสร้างของพอร์ต A (Pins PA0 – PA7)

พอร์ต B

พอร์ต B เป็นพอร์ตสองทิศทาง ขนาด 8 บิท ดดยมีรีจีสเตอร์ใช้ควบคุมพอร์ต B อยู่ 3 ตัว คือ รีจีสเตอร์ PORTB อยู่ที่ตำแหน่ง $18 ($38) รีจีสเตอร์ DDRB อยู่ที่ตำแหน่ง $17(&37) และ PINB อยู่ที่ตำแหน่ง $16($36) โดยพอร์ต A แต่ละขาสามารถแยกกำหนดให้มีความต้านทาน PULL UP ได้ตามต้องการ ซึ่งในแต่ละขาสามารถรับกระแส (SINK CURRENT) ได้ 20 mA โดยขาของพอร์ตสามารถใช้งานเป็นฟังชั่นอื่นๆ ได้ดังตารางที่ 2.11

ตารางที่ 2.11 การใช้งานฟังก์ชั่นอื่นๆ ของพอร์ต B

Port Pin	Alternate Function
PB0	T0 (Timer/Counter 0 external counter input)
PB1	T1 (Timer/Counter 1 external counter input)
PB2	AIN0 (Analog comparator positive input)
PB3	AIN1 (Analog comparator negative input)
PB4	SS (SPI Slave Select input)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB7	SCK (SPI Bus Serial Clock)

The Port B Data Register - PORTB

Bit	7	6	5	4	3	2	1	0	
$18 ($38)	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	

The Port B Data Direction Register - DDRB

Bit	7	6	5	4	3	2	1	0	
$17 ($37)	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	

The Port B Input Pins Address - PINB

Bit	7	6	5	4	3	2	1	0	
$16 ($36)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

การใช้งานพอร์ต B เป็นพอร์ตอินพุตและเอาท์พุต

จากตารางที่ 2.9 แสดงการใช้งานพอร์ต B เป็นอินพุตและเอาท์พุต

ตารางที่ 2.12 การใช้งานพอร์ต B เป็นอินพุตและเอาท์พุต

DDAn	PORTAn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PBn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

ฟังก์ชั่นอื่นๆ ที่ใช้ขาสัญญาณของ PORTB

SCK-PORTB,Bit 7

เป็นขา CLOCK ที่ใช้งานในส่วนของการสื่อสารแบบ SPI

MISO-PORTB,Bit6

เป็นขารับข้อมูลของการสื่อสารแบบ SPI

MOSI-PORTB,Bit5

เป็นขาส่งข้อมูลในของการสื่อสารแบบ SPI

SS-PORTB,Bit4

ใช้เป็นขาควบคุมการทำวานในโหมด SPI

AIN1-PORTB,Bit3

เป็นขาอินพุต 1 ของการทำงานในส่วนของ Analog Compare

AIN0-PORTB,Bit2

เป็นขาอินพุต 1 ของการทำงานในส่วนของ Analog Compare

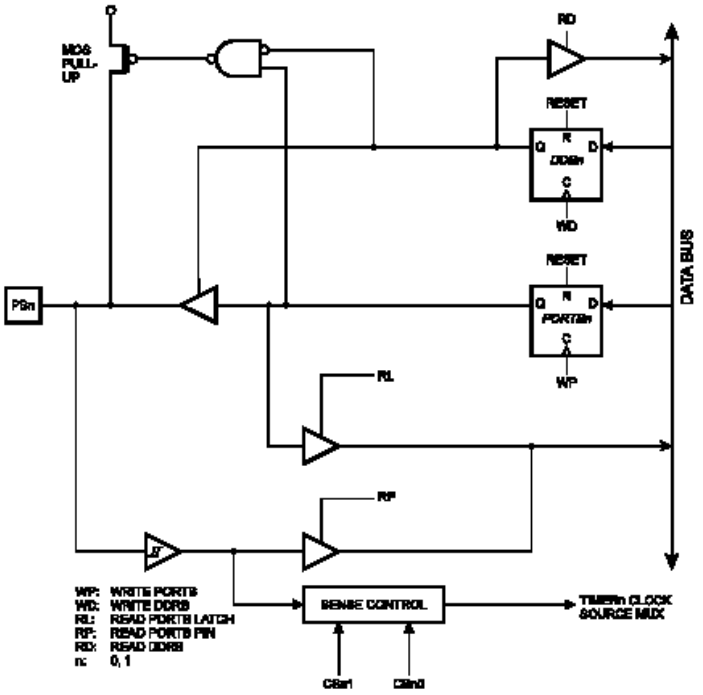
T1-PORTB,Bit 1

เป็นขาอินพุตที่ใช้ในส่วนของ Timer 1

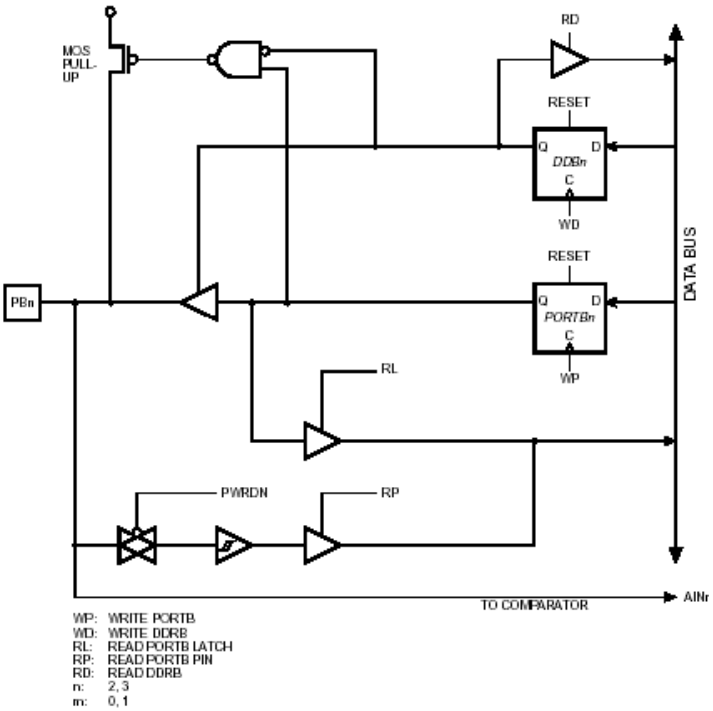
TO-PORTB,Bit0

เป็นขาอินพุตที่ใช้ในส่วนของ Timer 0

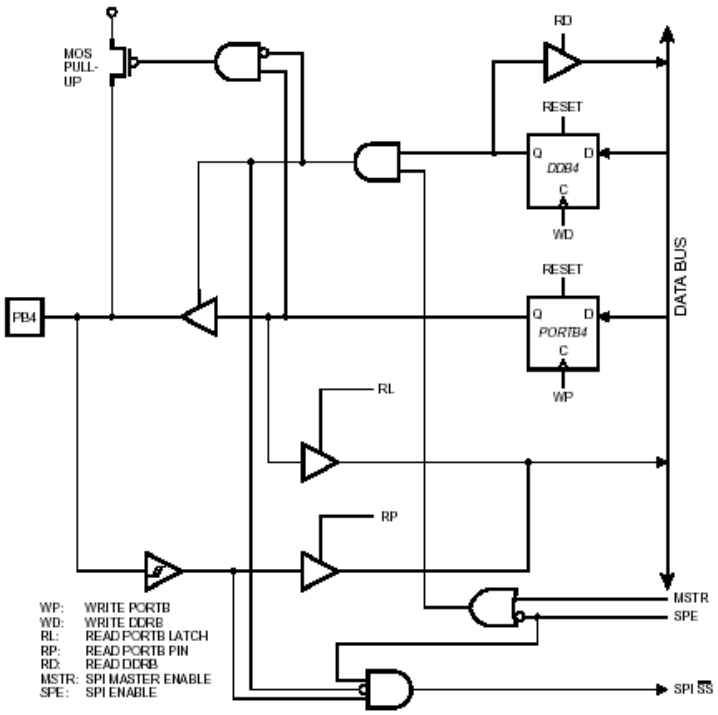
โครงสร้างของ PORT B



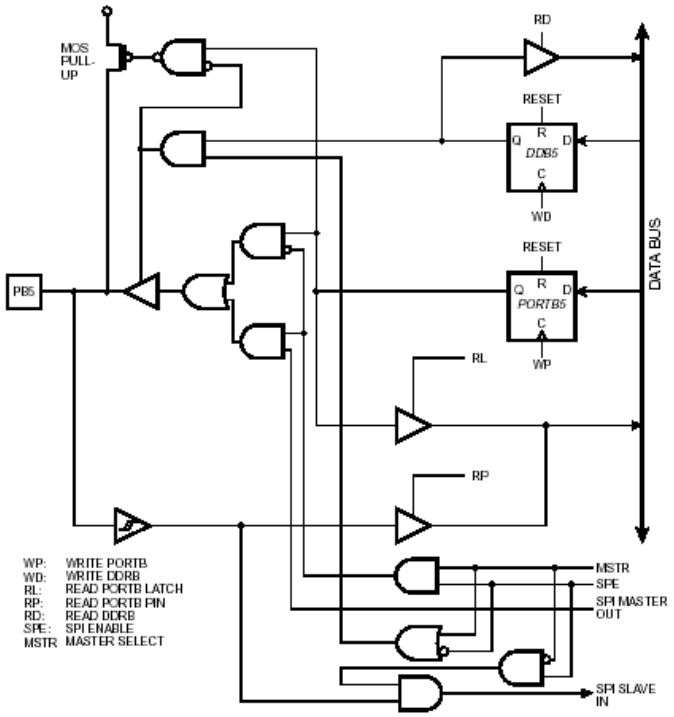
ภาพที่ 2.30 แสดงโครงสร้างของพอร์ต B (Pins PBO และ PB1)



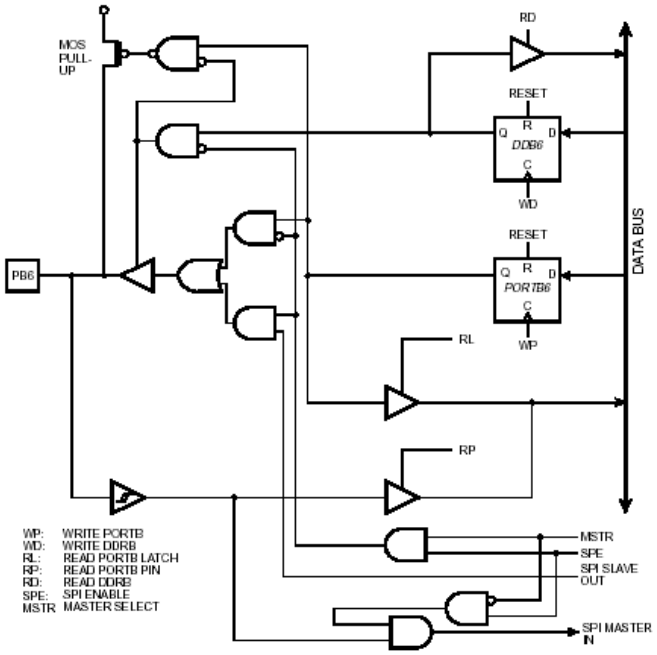
ภาพที่ 2.31 แสดงโครงสร้างของพอร์ต B (PB2 และ PB3)



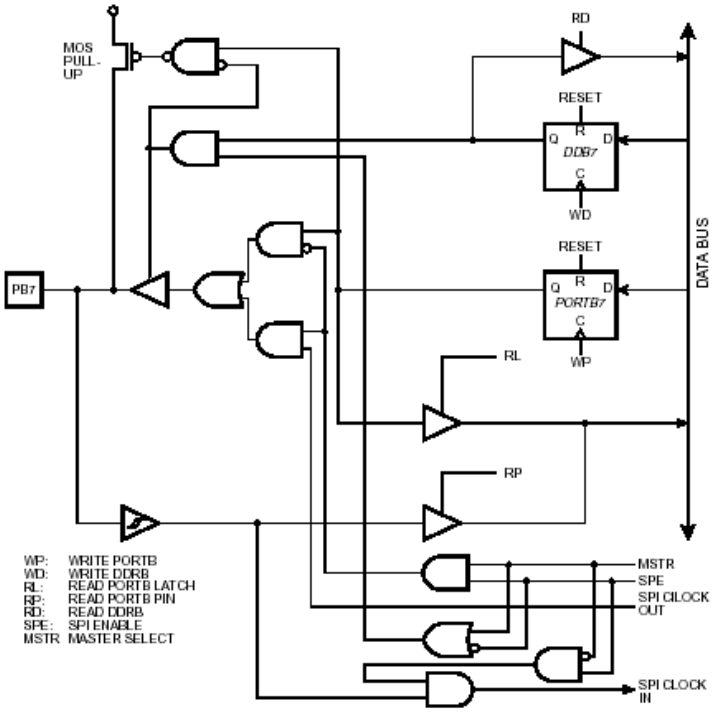
ภาพที่ 2.32 แสดงโครงสร้างของพอร์ต B (PB4)



ภาพที่ 2.33 แสดงโครงสร้างของพอร์ต B (PB5)



ภาพที่ 2.34 แสดงโครงสร้างของพอร์ตB (PB6)



ภาพที่ 2.35 แสดงโครงสร้างของพอร์ตB (PB7)

พอร์ต C

พอร์ต C เป็นพอร์ต 2 ทิศทางขนาด 8 บิท โดยมีหน่วยควบคุมการทำงานของพอร์ตจำนวน 3 หน่วย คือ PORTC(Data Register- PORTC) อยู่ที่ตำแหน่งหน่วยความจำ $15($35),DDRC (Data Direction Register DDRC) อยู่ที่ตำแหน่งหน่วย $14($34) และ PINC (Port C Input Pin - PINC) อยู่ที่ตำแหน่ง$13($)โดย PINC จะสามารถอ่านได้อย่างเดียว ในขณะที่ PORTC และ DDRC จะสามารถอ่านและเขียนได้โดยแต่ละขาของพอร์ต C สามารถแยกการกำหนดความต้านทาน PULL UP ได้ ในขณะที่พอร์ต C แต่ละขาสามารถรับกระแส(SINK CURRENT) ได้ 20 mA โดยถ้าภายในกำหนดให้มีความต้านทาน PULL UP และภายนอกมีการความต้านทาน PULL UP จะทำให้ MCU จ่ายกระแสออกภายนอก

The Port C Data Register - PORTC

Bit	7	6	5	4	3	2	1	0	
$15 ($35)	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	PORTC
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	

The Port C Data Direction Register - DDRC

Bit	7	6	5	4	3	2	1	0	
$14 ($34)	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
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	

The Port C Input Pins Address - PINC

Bit	7	6	5	4	3	2	1	0	
$13 ($33)	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

PINC ไม่ใช่ REGISTER เพราะฉะนั้นตำแหน่งที่กำหนดจะเป็นตำแหน่งของแต่ละขาของพอร์ต C โดยถ้าเป็นการอ่าน PORTC จะเป็นการอ่านค่าที่ LATCH ในขณะที่การอ่านค่าจาก PINC จะเป็นการอ่านค่าจริงของขาพอร์ต C

Port C As General Digital I/O

ทั้ง 8 บิท ของพอร์ต C จะมีลักษณะเหมือนกัน โดยPCn, แทนขาต่างๆ ของพอร์ต C และ DDCn เป็นบิทที่ใช้ควบคุมทิศทางของขานั้น ๆ โดยถ้า DDCn มีการเซ็ตเป็น 1 แสดงว่าเป็นขาเอาท์พุตแต่ถ้า DDCn ถูกเคลียเป็น 0 จะเป็นการกำหนดให้ขานั้นๆ เป็นขาอินพุต ถ้าต้องการให้ MOSS PULL เกิดการACTIVE จะเคลียให้ PORTCn เป็น 0 โดยลักษณะการกำหนดการใช้งานของแต่ละขาแสดงดังตารางที่ 2.10

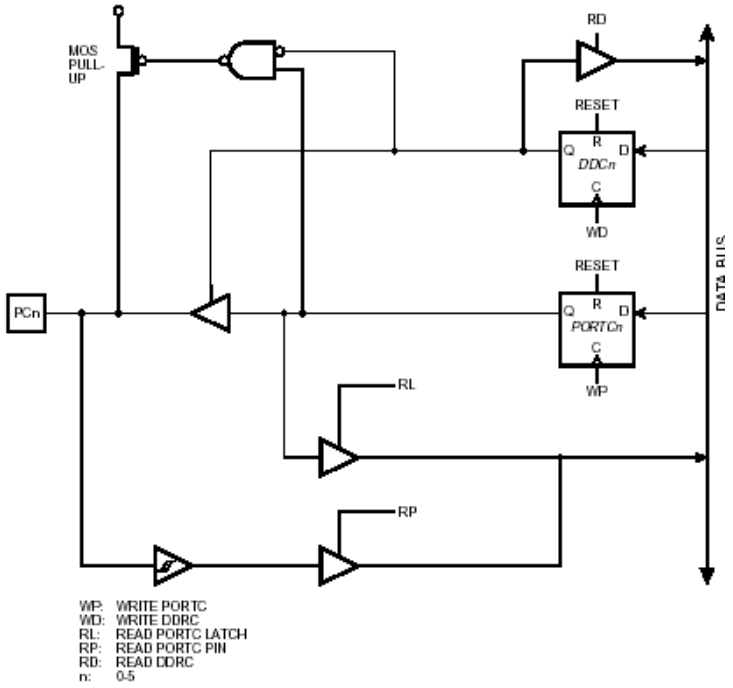
ตารางที่ 2.13 แสดงการใช้งานพอร์ต C

DDAn	PORTAn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PCn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

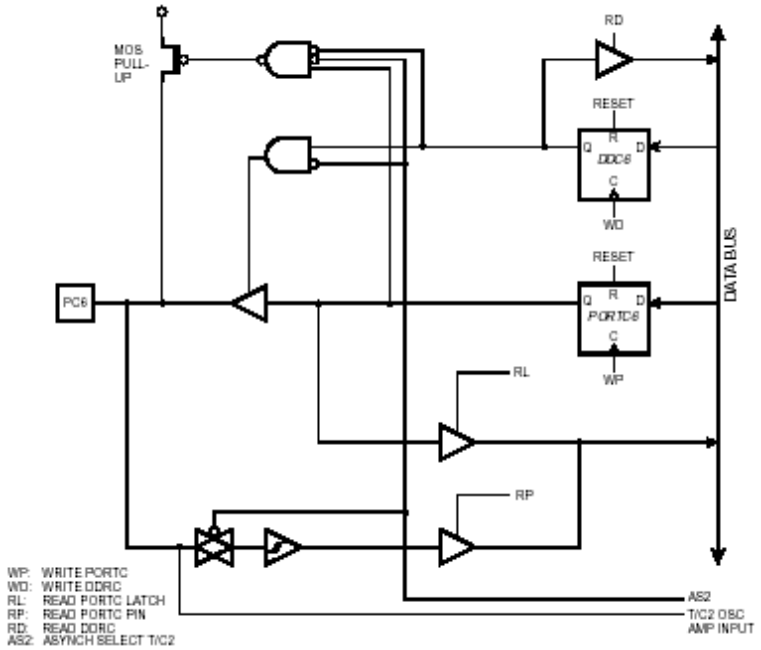
ฟังก์ชั่นอื่นๆ ของPORT C

โดยเมื่อบิท AS2 ในรีจีสเตอร์ ASSR ถูกกกเซ็ตเป็น 1 จะเป็นการกำหนดให้มีการรับ CLOCK จากภายนอกของ TIMER/COUNDTER2 ที่ตำแหน่งขาPC6,PC7

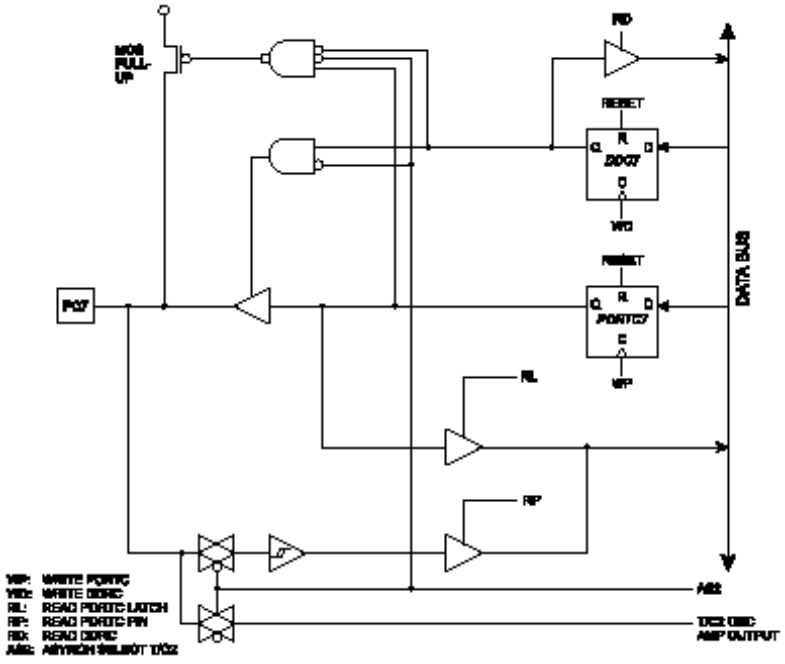
โครงสร้างของ PORT C



ภาพที่ 2.36 แสดงโครงสร้างของพอร์ต C(PCO-PC5)



ภาพที่ 2.37 แสดงโครงสร้างของพอร์ต C(PC6)



ภาพที่ 2.38 แสดงโครงสร้างของพอร์ต C (PC7)

พอร์ต D

พอร์ต D เป็นพอร์ต 2 ทิศทาง ขนาด 8 บิท ที่มีหน่วยควบคุมการทำงานของพอร์ต คือ PORTD (Data Register-PORTD) อยู่ที่ตำแหน่ง $12($32),DDRD (Data Direction Register) อยู่ที่ตำแหน่ง $11($31)และ PIND(Port D Input Pins) อยู่ที่ตำแหน่ง $10($30) โดยPIND สามารถอ่านได้อย่างเดียวในขณะที่PORTD และ DDRD สามารถทั้งอ่านและเขียน โดยพอร์ต D สามารถรับกระแสได้ 20 mA ซึ่งแต่ละขาของพอร์ต D สามารถเลือกฟังก์ชั่นการทำงานอื่นๆ ได้อีก

ตารางที่ 2.14 ฟังก์ชั่นอื่นๆ ของพอร์ต D

Port Pin	Alternate Function
PD0	RDX (UART Input line)
PD1	TDX (UART Output line)
PD2	INT0 (External interrupt 0 input)
PD3	INT1 (External interrupt 1 input)
PD4	OC1B (Timer/Counter1 output compareB match output)
PD5	OC1A (Timer/Counter1 output compareA match output)
PD6	ICP (Timer/Counter1 input capture pin)
PD7	OC2 (Timer/Counter2 output compare match output)

The Port D Data Register - PORTD

Bit	7	6	5	4	3	2	1	0	
$12 ($32)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
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	

The Port D Data Direction Register - DDRD

Bit	7	6	5	4	3	2	1	0	
$11 ($31)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
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	

The Port D Input Pins Address - PIND

Bit	7	6	5	4	3	2	1	0	
$10 ($30)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

PIND ไม่ใช่รีจีสเตอร์ ฉนั้นตำแหน่งที่กำหนดจะเป็นตำแหน่งจริงของขาพอร์ต โดยเมื่ออ่านค่าของPORT และ DDRD จะเป็นการอ่านค่าที่ LATCH ส่วนการอ่านค่าของ PIND จะเป็นการอ่านค่าจริง ๆ ที่เกิดขึ้นที่ขาของพอร์ต

การใช้งานของพอร์ต D เป็นพอร์ตอินพุตและเอาท์พุต

โดย PDn แทนขาใดๆ ของพอร์ต D และ DDDn เป็นตำแหน่งบิทในรีจีส-เตอร์ DDRD ซึ่งรีจีสเตอร์ DDRD ทำหน้าที่ควบคุมทิศทางของพอร์ต D โดยถ้าเซ็ตให้เป็น 1 จะเป็นการกำหนดให้พอร์ตทำงานเป็นเอาท์พุตแต่ถ้าเคลียให้เป็น 0 จะกำหนดให้พอร์ตทำงานเป็นอินพุต

ตารางที่ 2.15 แสดงการใช้งานพอร์ต D

DDAn	PORTAn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PDn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

การทำงานฟังก์ชั่นอื่นๆ ของพอร์ต D

OC2-PORTD,BIT7

ขา OC2 หรือบิท 7 ของพอร์ตD สามารถถูกนำไปใช้งานเป็นขาเอาท์พุตของ Compare Output Match ของ Time/Counter2

ICP-PORTD,BIT6

ขา ICP หรือบิท 6 ของพอร์ต D สามารถถูกนำไปใช้งานเป็นขาอินพุตของ Inprt Capture Mode ของ Timer/Counter1

OC1A-PORTD,BIT5

ขา OC1A หรือบิท 5 ของพอร์ต D สามารถถูกนำไปใช้งานเป็นขาเอาท์พุตของ Output CompareA Match ของ Timer/Counter1

OC1B-PORTD,BIT 4

ขาOC1B หรือบิท 4 ของพอร์ต D สามารถถูกนำไปใช้งานเป็นขาเอาท์พุตของ Output CompareB Match ของ Timer/Counter1

INT1-PORTD,BIT3

ขา INT1 หรือบิท 3ของพอร์ต D สามารถถูกใช้งานเป็นขาอินพุตของอินเตอร์รัพท์จากภายนอก

INT0-PORTD,BIT2

ขา INT0 หรือบิท 2 ของพอร์ต D สามารถถูกใช้งานเป็นขาอินพุตของอินเตอร์รัพท์จากภายนอก

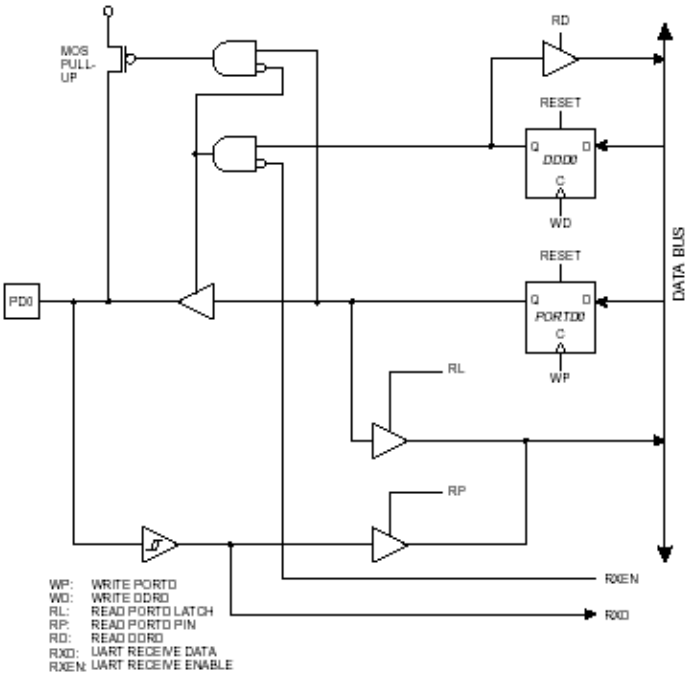
TXD-PORTD,BIT1

ขา TXD หรือบิท 1 ของพอร์ต D สามารถถูกใช้งานเป็นขาอินพุตส่งสัญญาณของการสื่อสารอนุกรมแบบ UART

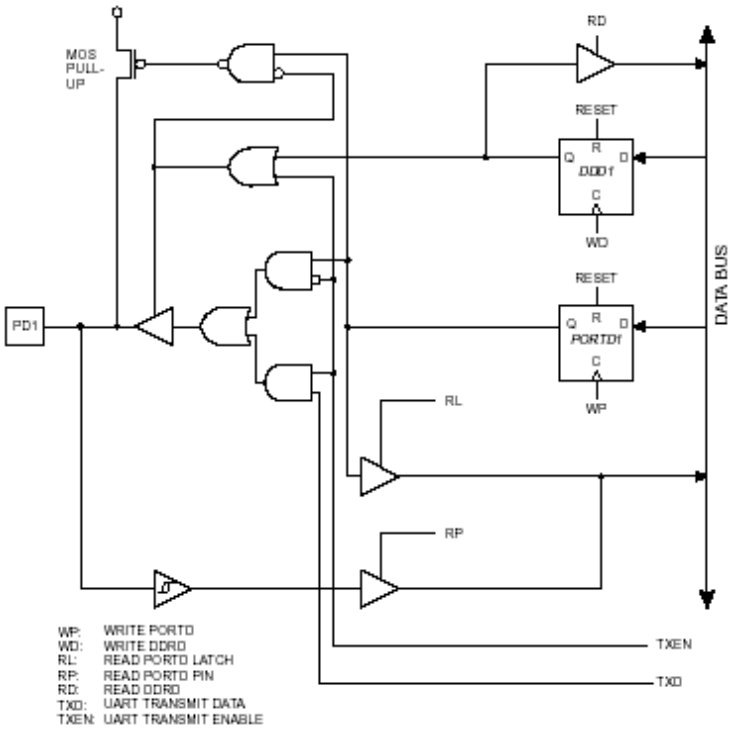
RXD-PORT,BIT0

ขา RXD หรือบิท 0 ของพอร์ต D สามารถถูกใช้งานเป็นขารับสัญญาณของการสื่อสารอนุกรมแบบ UART

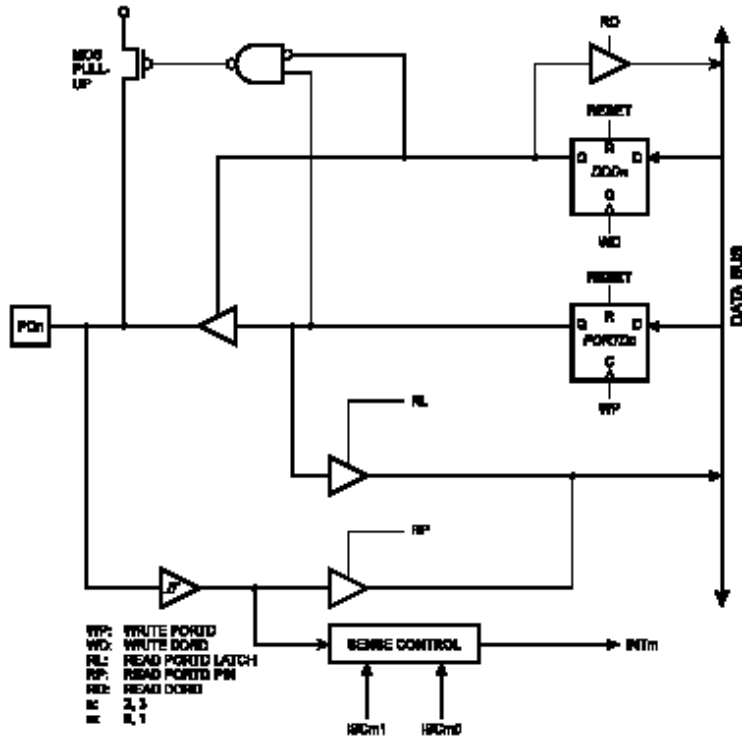
โครงสร้างของพอร์ต D



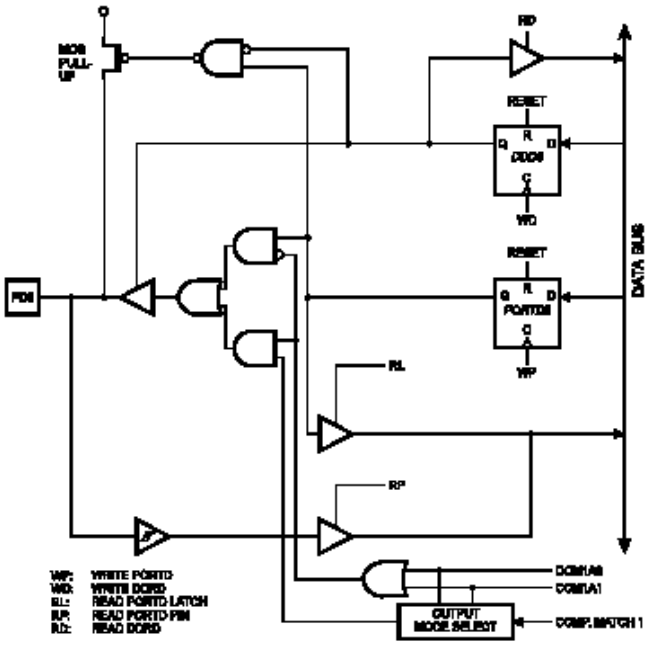
ภาพที่ 2.39 แสดงโครงสร้างของพอร์ต D(PO0)



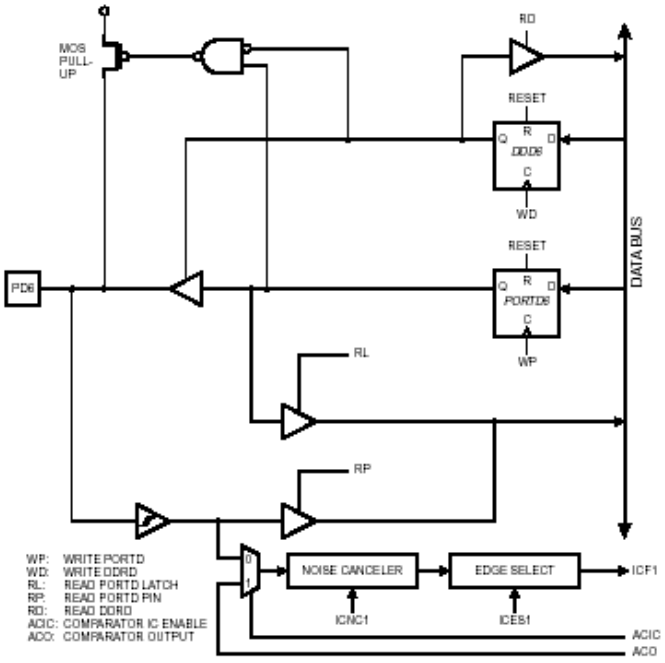
ภาพที่ 2.40 แสดงโครงสร้างของพอร์ต D(PD1)



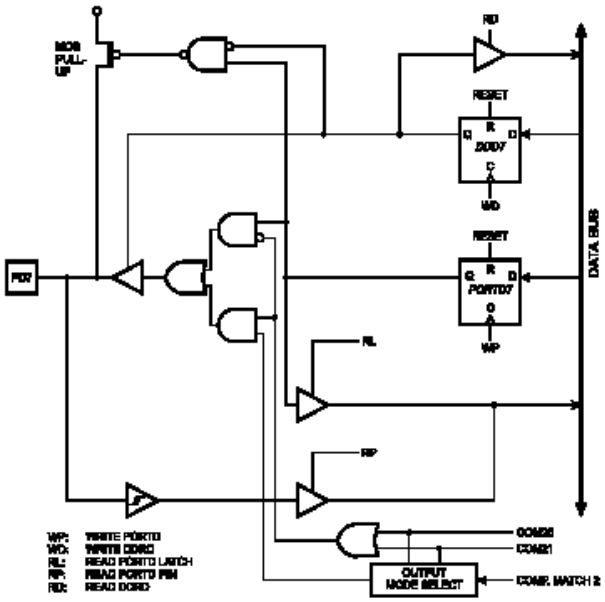
ภาพที่ 2.41 แสดงโครงสร้างของพอร์ต D(PD2และ PD3)



ภาพที่ 2.42 แสดงโครงสร้างของพอร์ต D (PD4 และ PD5)



ภาพที่ 2.43 แสดงโครงสร้างของพอร์ต D (PD6)



ภาพที่ 2.44 แสดงโครงสร้างของพอร์ต D (PD7)

กลุ่มคำสั่ง

กลุ่มคำสั่งการกระทำทางคณิตศาสตร์และลอจิก (ARITHMETIC AND LOGIC INSTRUCTION)

ADD Rd,Rr

คำสั่งบวกข้อมูลระหว่างรีจีสเตอร์ 2 ตัว ผลลัพท์ที่ได้เก็บไว้ในรีจีสเตอร์ Rd

ADC Rd,Rr

คำสั่งบวกข้อมูลระหว่างรีจีสเตอร์ 2 ตัว และ CARRY FLAG ผลลัพท์ที่ได้เก็บไว้ใน รีจีสเตอร์ Rd

ADIW Rdl,K

คำสั่งลบข้อมูลแบบ WORD ระหว่างรีจีสเตอร์กับ ค่าคงที่ ผลลัพท์ที่ได้เก็บในรีจีสเตอร์ Rdl

SUB Rd,Rr

คำสั่งลบข้อมูลระหว่างรีจีสเตอร์ 2 ตัว ผลลัพท์ที่ได้เก็บไว้ในรีจีสเตอร์ Rd

SUBI Rd,K

คำสั่งลบข้อมูลระหว่างรีจีสเตอร์กับค่าคงที่ ผลลัพท์ที่ได้เก็บไว้ในรีจีสเตอร์ Rd

SBC Rd,Rr

คำสั่งลบข้อมูลระหว่างรีจีสเตอร์ 2 ตัว และ CARRY FLAG ผลลัพท์ที่ได้เก็บไว้ในรีจีส-เตอร์ Rd

SBCI Rd,K

คำสั่งในการลบข้อมูลระหว่างรีจีสเตอร์กับค่าคงที่ และ CARRY FLAG ผลลัพท์ที่ได้เก็บไว้ในรีจีสเตอร์ Rd

SBIW Rdl,K

คำสั่งในการลบข้อมูลขนาด WORD

AND Rd,Rr

คำสั่ง AND ข้อมูลระหว่างรีจีสเตอร์ 2 ตัว

ANDI Rd,K

คำสั่ง AND ข้อมูลระหว่างรีจีสเตอร์ กับค่าคงที่

OR Rd,Rrฃ

คำสั่ง OR ข้อมูลระหว่างรีจีสเตอร์ 2 ตัว

ORI Rd,K

คำสั่ง OR ข้อมูลระหว่างรีจีสเตอร์กับค่าคงที่ ผลลัพท์ที่ได้เก็บไว้ในรีจีสเตอร์ Rd

EOR Rd,Rr

คำสั่ง Exclusive or ข้อมูลระหว่างรีจีสเตอร์ 2 ตัว

COM Rd

คำสั่งทำ one Complement ข้อมูลในรีจีสเตอร์ Rd

NEC Rd

คำสั่งทำ two Complement ข้อมูลในรีจีสเตอร์ Rd

SBR Rd,K

คำสั่ง SET บิทในรีจีสเตอร์ Rd

CBR Rd,K

คำสั่งเคลียร์บิทในรีจีสเตอร์ Rd

INC Rd

คำสั่งเพิ่มค่าของข้อมูลในรีจีสเตอร์ Rd ขึ้น 1 ค่า

DEC Rd

คำสั่งลดค่าของข้อมูลในรีจีสเตอร์ Rd ลง 1 ค่า

TST Rd

คำสั่งทดสอบข้อมูลในรีจีสเตอร์ Rd โดยจะมีผลกับ Flag Z,N,V

CLR Rd

คำสั่งเคลียร์ข้อมูลในรีจีสเตอร์ Rd ให้มีค่าเป็น 00

SER Rd

คำสั่ง SET ข้อมูลในรีจีสเตอร์ Rd ให้มีค่าเป็น FF

กลุ่มคำสั่งการกระโดด (BRANCH INSTRUCTION)

RJMP k

คำสั่งกระโดดแบบ Relative

IJMP

คำสั่งกระโดดไปยังตำแหน่งหน่วยความจำที่ถูกชี้โดยรีจีสเตอร์ Z

RCALL k

คำสั่งกระโดดไปยังโปรแกรมย่อยที่ k

ICALL

คำสั่งกระโดดไปทำงานโปรแกรมย่อยที่ชี้ตำแหน่งโดยรีจีสเตอร์ Z

RET

คำสั่งกลับจากโปรแกรมย่อย

RETI

คำสั่งกลับจากโปรแกรมบริการอิน เตอร์รัพท์

CPSE Rd,Rr

คำสั่งเปรียบเทียบข้อมูลใน Register Rd กับ ข้อมูลใน Register Rr โดยถ้าข้อมูลใน Register ทั้ง 2 มีค่าเท่ากัน CPU ข้ามการทำคำสั่งถัดไป 1 คำสั่ง

CP Rd,Rr

คำสั่งเปรียบเทียบข้อมูลในรีจีสเตอร์ Rd และข้อมูลในรีจีสเตอร์ Rr โดยจะมีผลต่อ Flag Z,N,V,C,H

CPC Rd,Rr

คำสั่งเปรียบเทียบข้อมูลในรีจีสเตอร์ Rd กับข้อมูลในรีจีสเตอร์ Rr และ Carry Flag โดยมีกากรกระทำ Rd-Rr-C ซึ่งทำให้มีผลกับ Flag Z,N,V,C,H

CPI Rd,k

คำสั่งเปรียบเทียบข้อมูลในรีจีสเตอร์ Rd กับค่าคงที่ โดย CPU จะกระทำ Rd-k ซึ่งมีผลกับ flag Z,N,V,C,H

SBRC Rr,b

เมื่อ CPU ทำคำสั่ง SBRC จะทำให้ CPU จะข้ามการกระทำคำสั่งถัดไป 1 คำสั่ง ถ้าบิทที่ b ในรีจีสเตอร์ Rr มีค่าเป็น 0

SBRS Rr,b

เมื่อ CPU ทำคำสั่ง SBRS จะทำให้ CPU ข้ามการกระทำคำสั่งถัดไป 1 คำสั่ง ถ้าบิทที่ b ในรีจีสเตอร์ Rr ถูกเซ็ค

SBIC P,b

เมื่อ CPU ทำคำสั่ง SBIC จะทำให้ CPU ข้ามการทำคำสั่งถัดไป 1 คำสั่ง ถ้าบิทที่ b ในรีจีสเตอร์ I/O ที่ P ถูกเครียร์

SBIS P,b

เมื่อ CPU ทำคำสั่ง SBIS จะทำให้ CPU ข้ามการทำคำสั่งถัดไป 1 คำสั่ง ถ้าบิทที่ b ในรีจีสเตอร์ I/O ที่ P ถูกเซ็ต

BRBS s,k

เป็นคำสั่งกระโดดข้ามไปทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Flag Status ในรีจีสเตอร์ SREG ถูกเซ็ต (SREG(s)) = 1 โดยค่าใน PC = PC+k+1

BRBC s,k

เป็นคำสั่งกระโดดข้ามไปทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Flag Status ในรีจีสเตอร์ SREG ถูกเคลียร์ (SREG(s)) = 0 โดยค่าใน PC = PC+k+1

BREQ k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่ง PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRNE k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRCS k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRCC k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRSH k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRLO k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRMI k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRPL k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRGE k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRLT k

เป็นคำสั่งกระโดดข้ามไปกระทำคำสั่งในตำแหน่งที่ PC ชี้ ถ้า Zero flag เซ็ต (Z=1) โดยค่าใน PC = PC+k+1

BRSH k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า

BRHC k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า Half Carry เซ็ต

BRTS k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า T flag Set

BRTC k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า T flag Clear

BRVS k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า Overflow Flag Set

BRVC k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า Overflow Flag ถูกเคลียร์

BRIE k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า Interrupt Enable บิท (I=1)

BRID k

คำสั่งกระโดดไปยังหน่วยความจำตำแหน่ง PC = PC+k+1 ถ้า Interrup Disable บิท (I=0)

กลุ่มคำสั่งในการเคลื่อนย้ายข้อมูล (DATA TRANFER INSTRUCTIONS)

MOV Rd,Rr

เป็นคำสั่งในการเคลื่อนย้ายข้อมูลจาก รีจีสเตอร์ ไปยังรีจีสเตอร์

Rd คือ รีจีสเตอร์ RO – R31

Rr คือ รีจีสเตอร์ RO – R31

LDI Rd,K

เป็นคำสั่งในการโหลดข้อมูลคงที่ขนาด 8 บิท เข้าในรีจีสเตอร์

Rd คือ รีจีสเตอร์ R16 – R31

K คือ ค่าคงที่ขนาด 8 บิท O – 255

LD Rd,X

เป็นคำสั่งในการโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่ รีจีสเตอร์ X ชี้ เข้าในรีจีสเตอร์

Rd คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

LD Rd,X+

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ X ชี้ เข้าในรีจีสเตอร์ทั่วไป และเพิ่มค่ารีจีสเตอร์ X ขึ้น 1 หลังจากที่โหลดข้อมูลเสร็จแล้ว

Rd คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

LD Rd,X-

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ X –1 ชี้ เข้าในรีจีสเตอร์ทั่วไป และลดค่า X ลง 1 ค่าก่อนที่จะโหลดข้อมูลลง

Rd คือ รีจีสเตอร์ RO –R31

X คือ รีจีสเตอร์ X

LD Rd,X

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ X ชี้ เข้าในรีจีสเตอร์

Rd คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

LD Rd,X+

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ X ชี้เข้าในรีจีสเตอร์ทั่วไปและเพิ่มค่ารีจีสเตอร์ X ขึ้น 1 ค่าหลังจากที่โหลดข้อมูลเสร็จแล้ว

Rd คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

LD Rd,-X

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิท ที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ X-1 ชี้ เข้าในรีจีสเตอร์ทั่วไป โดยจะทำการลดค่า X ลง 1 ค่า ก่อนที่จะโหลดข้อมูลลง

Rd คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

LD Rd,Y

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y ชี้ เข้าในรีจีสเตอร์

Rd คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

LD Rd,Y+

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y ชี้เข้าในรีจีสเตอร์ทั่วไปและเพิ่มค่ารีจีสเตอร์ Y ขึ้น 1 ค่าหลังจากที่โหลดข้อมูลเสร็จแล้ว

Rd คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

LD Rd,-Y

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y-1 ชี้เข้าในรีจีสเตอร์ทั่วไปโดยจะทำการลดค่า Y ลง 1 ค่า ก่อนที่จะโหลดข้อมูลลง

Rd คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

LD Rd,Y-q

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิท ที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y+q ชี้ เข้าไปในรีจีสเตอร์ทั่วไป

Rd คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

Q คือ ค่าระยะห่าง มีค่า 0-63

LD Rd,Z

เป็นค่าคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Z ชี้ เข้าในรีจีสเตอร์

Rd คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

LD Rd, Z+

เป็นค่าคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Z ชี้เข้าในรีจีสเตอร์ทั่วไปและเพิ่มค่ารีจีสเตอร์ Z ขึ้น 1 ค่า หลังจากที่โหลดข้อมูลเสร็จแล้ว

Rd คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

LD Rd,-Z

เป็นค่าคำสั่งโหลดข้อมูลขนาด 8 บิทที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Z-1 ชี้ เข้าในรีจีสเตอร์ทั่วไป โดยจะทำการลดค่า Z ลง 1 ค่า ก่อนที่จะโหลดข้อมูลลง

Rd คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

LD Rd,Z+q

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิท ที่อยู่ในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Z+q ชี้ เข้าไปในรีจีสเตอร์ทั่วไป

Rd คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

q คือ ค่าระยะห่าง มีค่า 0-63

LDS Rd,k

เป็นคำสั่งโหลดข้อมูลขนาด 8 บิท ที่อยู่ในหน่วยความจำ SRAM ในตำแหน่งที่ k เข้าในรีจีสเตอร์ทั่วไป

Rd คือ รีจีสเตอร์ RO – R31

K คือ ตำแหน่งหน่วยความจำใน SRAM

ST X,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่ รีจีสเตอร์ X ชี้

Rr คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

ST X+,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่ รีจีสเตอร์ X ชี้ และทำการเพิ่มค่ารีจีสเตอร์ X ขึ้น 1 ค่า

Rr คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

ST -X,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่รีจีสเตอร์ X-1 ชี้

Rr คือ รีจีสเตอร์ RO – R31

X คือ รีจีสเตอร์ X

ST Y,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y ชี้

Rr คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

ST Y+, Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y ชี้และเพิ่มค่ารีจีสเตอร์ Y ขึ้น 1 ค่า

Rr คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

ST -Y,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y-1 ชี้

Rr คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

ST Y+q,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Y+q ชี้

Rr คือ รีจีสเตอร์ RO – R31

Y คือ รีจีสเตอร์ Y

ST Z,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่ รีจีสเตอร์ Z ชี้

Rr คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

ST Z+,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่ รีจีสเตอร์ Z ชี้ และเพิ่มค่ารีจีสเตอร์ Z ขึ้น 1 ค่า

Rr คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

ST -Z,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ทั่วไปลงในหน่วยความจำตำแหน่งที่ รีจีสเตอร์ Z –1 ชี้

Rr คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

ST Z+q,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ใช้งานทั่วไปลงในหน่วยความจำตำแหน่งที่รีจีสเตอร์ Z +q ชี้

Rr คือ รีจีสเตอร์ RO – R31

Z คือ รีจีสเตอร์ Z

Q คือ ค่าระยะห่าง 0-63

STS k,Rr

เป็นคำสั่งบันทึกข้อมูลขนาด 8 บิท จากรีจีสเตอร์ใช้งานทั่วไปลงในหน่วยความจำตำแหน่งที่ k

Rr คือ รีจีสเตอร์ RO – R31

K คือ ตำแหน่งในหน่วยความจำ 0-65535

LPM

โหลด Program Memory

IN Rd,P

คำสั่งอ่านข้อมูลจากรีจีสเตอร์ I/O มาเก็บไว้ในรีจีสเตอร์ทั่วไป

Rd คือ รีจีสเตอร์ RO – R31

P คือ รีจีสเตอร์ อินพุท/เอาท์พุท

OUT P,Rd

คำสั่งเขียนข้อมูลจากรีจีสเตอร์ทั่วไปเข้าไปในรีจีสเตอร์อินพุทเอาท์พุท

Rd คือ รีจีสเตอร์ RO – R31

P คือ รีจีสเตอร์ อินพุท/เอาท์พุท

PUSH Rr

คำสั่งเก็บค่ารีจีสเตอร์ Rr ลงใน STACK

POP Rr

คำสั่งย้ายค่าจาก STACK มาเก็บในรีจีสเตอร์ทั่วไป

กลุ่มคำสั่งการกระทำทางบิทและการสอบบิท

SBI P,b

คำสั่งเซ็ต bit b ในรีจีสเตอร์ I/O ตำแหน่งที่ P

I/O(P,b) 1

CBI P,b

คำสั่งเคลียร์บิท b ในรีจีสเตอร์ I/O ตำแหน่งที่ P

I/O(P,b) 0

LSL Rd

เลื่อนข้อมูลในรีจีสเตอร์ Rd ไปทางซ้าย 1 บิท

LSR Rd

เลื่อนข้อมูลในรีจีสเตอร์ Rd ไปทางขวา 1 บิท

ROL Rd

หมุนข้อมูลไปทางซ้าย 1 บิท โดยผ่าน Carry flag

ROR Rd

หมุนข้อมูลไปทางขวา 1 บิท โดยผ่าน Carry flag

SWAP Rd

คำสั่งสลับค่า 4 bit high และ 4 bit low ใน register Rd

BSET s

เซ็ต Flag

BCLR s

เคลียร์ Flag

BST Rr,b

คำสั่งโหลดบิท T ลงในบิท b ของรีจีสเตอร์ Rr

BLD Rr,b

คำสั่งโหลดบิท b ในรีจีสเตอร์ Rr เก็บไว้ในบิท T

SEC

คำสั่งเซ็ต carry flag

CLC

คำสั่งเคลียร์ carry flag

SEN

คำสั่งเซ็ต negative flag

CLN

คำสั่งเคลียร์ negative flag

SEZ

คำสั่งเซ็ต zero flag

CLZ

คำสั่งเคลียร์ zero flag

SEI

คำสั่ง Enable Gobal Interrupt

CLI

คำสั่ง Disable Interrupt ทั้งหมด

SES

คำสั่งเซ็ต signed test flag

CLS

คำสั่งเคลียร์ signed test flag

SEV

คำสั่งเซ็ต 2's complement Overflow

CLV

คำสั่งเคลียร์ 2's complement Overflow

SET

คำสั่งเซ็ตบิท T ในรีจีสเตอร์ SREG

CLT

คำสั่งเคลียร์บิท T ในรีจีสเตอร์ SREG

SEH

คำสั่งเซ็ต half carry flag ในรีจีเตอร์ SREG

NOP

คำสั่ง NO OPERATION

SLEEP

คำสั่งให้ CPU ทำงาน SLEEP MODE

WDR

คำสั่งรีเซ็ต Watchdog