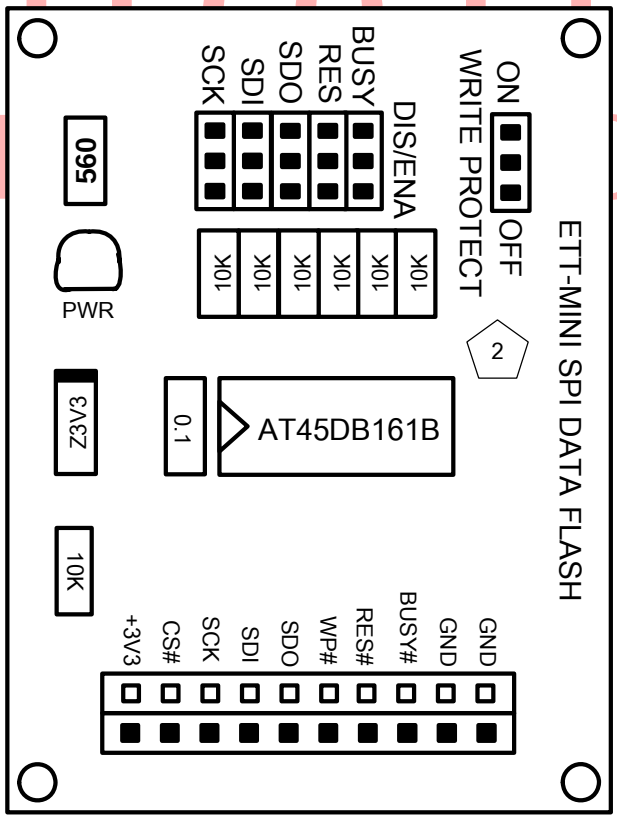
ET-MINI SPI DATA FLASH

Module ET-MINI SPI DATA FLASH เป็นชุดสำหรับเก็บข้อมูล(Data Memory Flash #AT45DB161B) โดยจะ Interface Read/Write ข้อมูล แบบ Serial SPI โดย Module จะมีคุณสมบัติดังนี้

1. ใช้กับ Power Supply 2.5V-3.6V เท่านั้น
2. ขาสัญญาณ Input SI , SCK , $\overline{CS}$, $\overline{RESET}$ และ $\overline{WP}$ สามารถรองรับแรงดัน 5.0V ได้
3. ใช้การ Interface แบบ SPI ที่รองรับความถี่ Clock ได้สูงสุด 20 Mhz
4. สามารถเก็บข้อมูลได้ถึง 2 MB
5. มีขาสำหรับป้องกันการเขียนข้อมูลทับลงในหน่วยความจำ
6. หน่วยความจำนี้ จะถูกแบ่งออกเป็น 4096 Page และในแต่ละ Page จะมีขนาด 528 Byte
7. สามารถลบข้อมูลเป็น Page(528 Byte) หรือเป็น Block(1Block= 8 Page)ได้
8. มี SRAM Data Buffer อยู่ภายใน 2 ชุด ซึ่งมีขนาด 528 Byte เพื่อทำหน้าที่รับข้อมูลไปพักไว้ ก่อนที่จะส่งไปยังหน่วยความจำหลัก(พื้นที่ Flash) ซึ่งจะขึ้นอยู่กับ Command ที่ใช้ ว่าจะให้ผ่าน Buffer หรือจะเข้าถึง หน่วยความจำหลักโดยตรง



รูปที่1 Module ET-MINI SPI DATA FLASH

หน้าที่ของขาสัญญาณ

- CS# (Chip select) : Data Flash จะถูกเลือกเมื่อขา $\overline{CS}$ นี้เป็น Low และถ้าขานี้เป็น High จะไม่มีการส่งข้อมูลออกไปที่ขา SI ส่วนขา SO จะยังอยู่ในสถานะ High-impedance เมื่อมีการส่งสัญญาณ จาก High- to-Low ไปยังขา $\overline{CS}$ นั่นหมายถึงต้องการเริ่มต้นการทำงาน และถ้าส่งสัญญาณ จาก Low-to-High ไปยังขา $\overline{CS}$ อีกครั้ง จะเป็นการจบการทำงาน

-SCK (Serial Clock) : ขานี้จะทำหน้าที่เป็น Input เพื่อรับสัญญาณ Clock จากภายนอกเข้ามาควบคุมจังหวะการไหลเข้าออกของข้อมูลจาก Data Flash

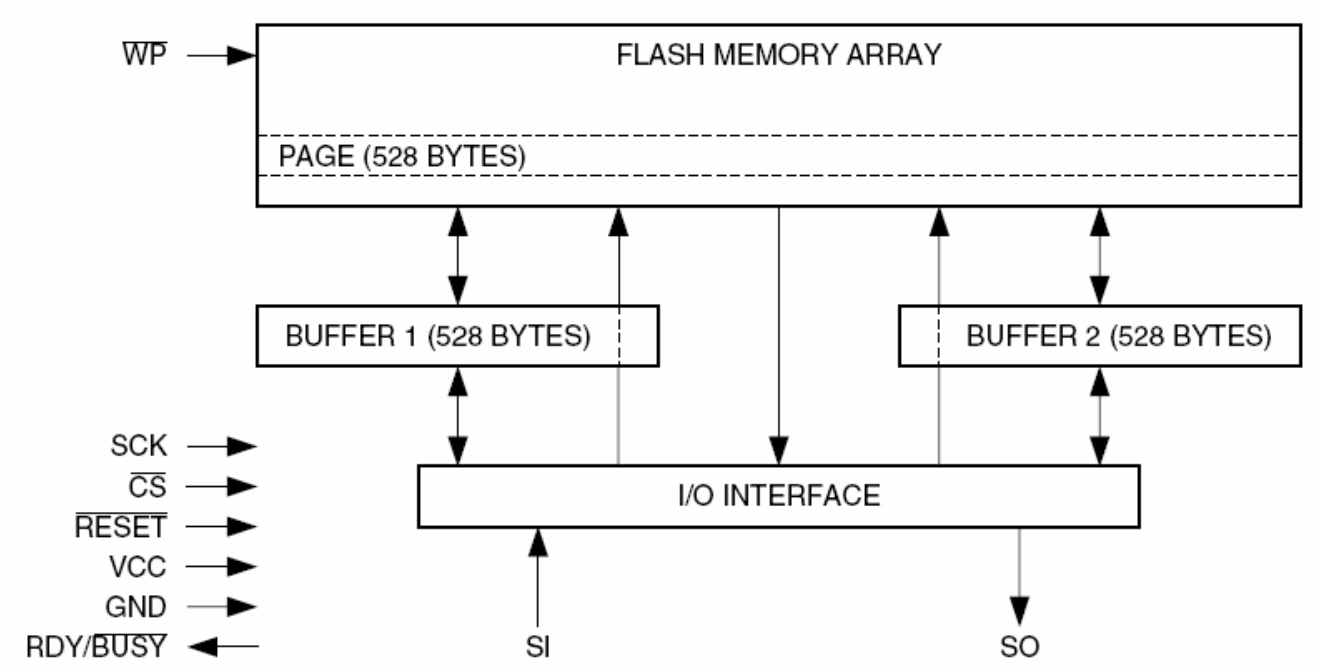
-SI (Serial Input) : ขานี้จะทำหน้าที่เป็น Input และถูกใช้เพื่อเลื่อนข้อมูลเข้าไปยัง Chip หน่วยความจำ

-SO (Serial Output) : ขานี้จะทำหน้าที่เป็น Output ถูกใช้เพื่อเลื่อนข้อมูลออกจากตัว Chip หน่วยความจำ

-WP# (Write Protect) : ถ้าขานี้เป็น Low จะทำให้ในพื้นที่หน่วยความจำหลัก 256 Page แรก ไม่สามารถที่จะเขียนข้อมูลเข้าไปเก็บไว้ได้

-RES# (Reset) : เมื่อขานี้เป็น Low จะทำให้ Chip หน่วยความจำถูก Reset กระบวนการทำงานต่างๆก็จะสิ้นสุดลง และจะทำงานได้ใหม่เมื่อขานี้เป็น High

-BUSY# (Ready/Busy) : ขานี้จะทำหน้าที่เป็น Open drain Output ซึ่งจะให้สัญญาณเป็น Low เมื่อการทำงานภายในของ Chip หน่วยความจำยังไม่พร้อม ในสภาวะปกติขานี้จะต้องเป็น High

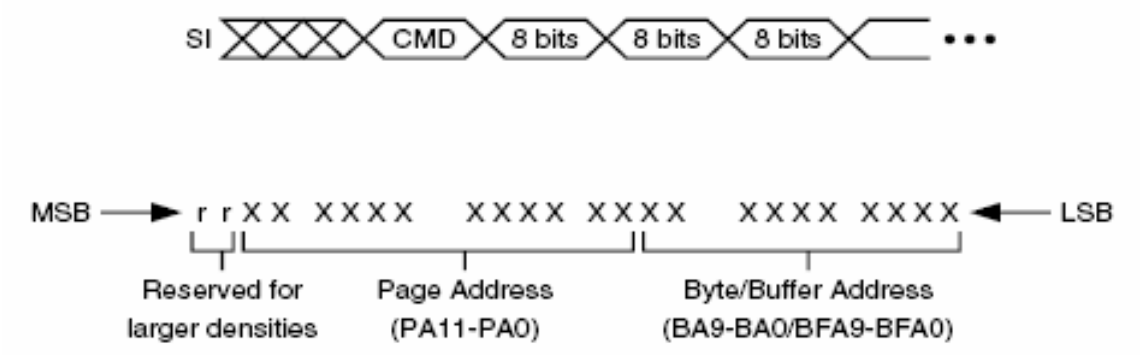


รูปที่2 Block Diagram Memory Chip #AT45DB161B

สำหรับพื้นที่หน่วยความจำหลักของ Chip หน่วยความจำ #AT45DB161B จะถูกแบ่งออกเป็น 3 ระดับ ซึ่งประกอบไปด้วย Sector(มี 32 Block/Sector) , Block(มี 8Page/Block) และ Page(มี528Byte/Page) การทำงานของโปรแกรม ในการ Read/Write data Flash นั้น จะปรากฏในลักษณะของ Page-by-Page คือเวลาที่จะ Read/Write ข้อมูลแต่ละครั้งจะต้องทำครั้งละ 1 Page หรือ 528 Byte ส่วนในการลบข้อมูลนั้น สามารถลบได้ทั้งแบบ Block หรือแบบ Page

การทำงานของ Chip Memory Flash

การทำงานของอุปกรณ์นี้จะถูกควบคุมด้วยคำสั่ง ที่ส่งมาจาก MCU ผ่านทางการสื่อสารแบบ SPI โดย Opcode ของคำสั่งจะแสดงดังในตารางที่ 1-4 ด้านล่าง คำสั่งจะเป็น Opcode ขนาด 8 บิต จะถูก Start ที่ขอบขาลงของสัญญาณ $\overline{CS}$ ขณะที่ $\overline{CS}$ เป็น Low และมีสัญญาณ Clock เข้ามาก็จะทำให้เกิดการโหลด Opcode และตำแหน่ง Address ของ Buffer หรือ Address ของหน่วยความจำหลักออกไปยังขา SI โดยจะส่งบิต MSB ออกไปเป็นบิตแรก รูปแบบในการส่ง Command Read/Write จะแสดงดังรูปด้านล่าง



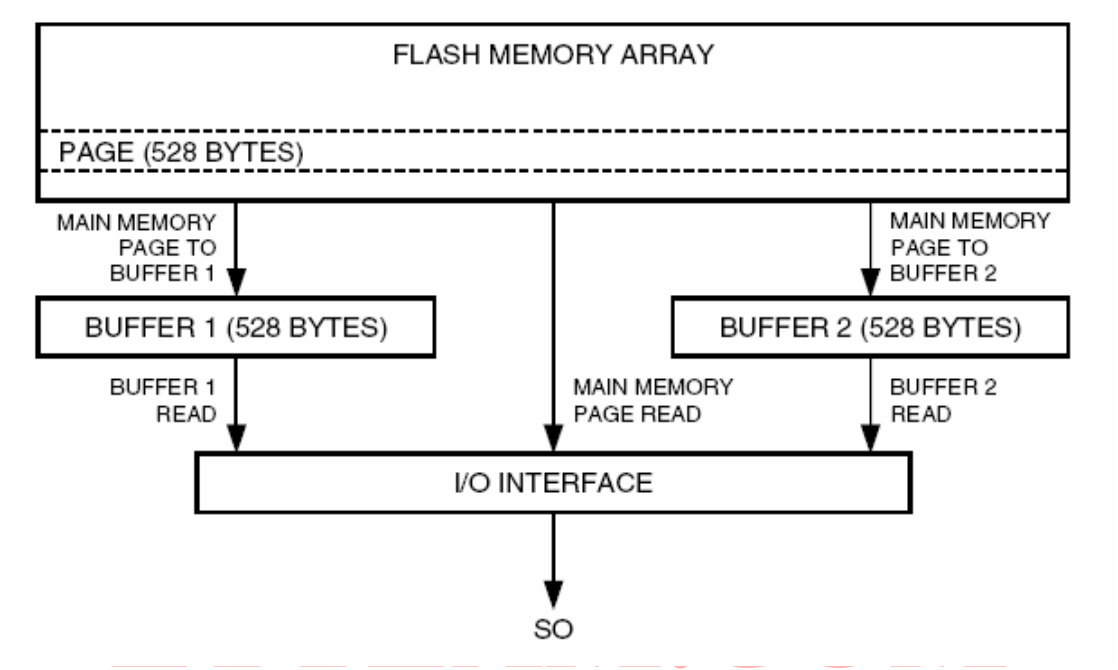
รูปที่3 แสดง รูปแบบการส่ง Command สำหรับ Read/Write

ในการส่ง Command แต่ละครั้งผู้ใช้จะต้องเรียงลำดับการส่งให้เป็นไปตามรูปแบบด้านบนเสมอ โดย Byte แรกจะต้องส่ง Opcode ของ Command ออกไป จากนั้นก็ตามด้วย Address 3 Byte(24bit) ที่เหลือโดยแยกรายละเอียดให้เห็นในรูปด้านบน ซึ่งประกอบด้วย rr (2bit)2บิตนี้จะถูกสงวนไว้ใช้กับความจุของ Chip ที่มีขนาดมากขึ้น ดังนั้นจะต้องกำหนดให้เป็น ‘00’ ส่วนอีก 12 บิตต่อมา จะใช้กำหนดตำแหน่งของ Page Address(PA11-PA0)ซึ่งจะสามารถอ้าง Address ได้ตั้งแต่ค่า 0-4095 และ 10 บิต สุดท้ายของ Command ที่จะต้องส่งออกไปคือ Byte/Buffer Address (BA9-BA0/BFA9-BFA0) ซึ่งก็คือตำแหน่ง Address ภายใน Page นั้นๆ หรือ ตำแหน่ง Address ใน Buffer1 หรือ Buffer2 ซึ่งจะสามารถอ้าง Address ได้ตั้งแต่ค่า 0-527

ในการใช้งานแต่ละคำสั่งนั้น บางคำสั่งจะอ้างเพียง Page Address หรือ อ้างเพียง Byte/ Buffer Address อย่างใดอย่างหนึ่งเท่านั้น ในส่วนที่ไม่ได้อ้างถึงนั้นผู้ใช้จะต้องใส่ค่า Don’t Care (0 หรือ 1 ก็ได้) แทนในส่วน ที่ไม่ได้อ้างถึงเพื่อให้ Command ที่ส่งออกไปครบ 4 Byte ซึ่งสามารถดูรายละเอียดการส่ง Command แต่ละบิตได้ในตารางที่ 4

การอ่านและคำสั่งที่ใช้ในการอ่านข้อมูล

การอ่านข้อมูลจาก Chip นั้นจะแสดงให้เห็นดังบล็อกไดอะแกรมด้านล่าง ซึ่งจะสอดคล้องกับคำสั่งที่ใช้ในการอ่านดังตารางที่1 จากบล็อกไดอะแกรมจะเห็นว่าการอ่านข้อมูลนั้นมีการอ่านได้หลายแบบ ซึ่งในแต่ละแบบคำสั่งที่ใช้ในการอ่านก็จะแตกต่างกันไปผู้ใช้จึงต้องทำความเข้าใจในแต่ละคำสั่งเสียก่อนเพื่อจะเลือกใช้คำสั่งได้ถูกต้อง คำสั่งที่ใช้ในการอ่านนั้นอาทิเช่น อ่านข้อมูลจาก Flash ไปเก็บไว้ที่ Buffer 1 หรือ 2 ภายในตัว Chip เอง จากนั้นจึงใช้คำสั่งอ่านข้อมูลจาก Buffer 1 หรือ 2 เพื่อส่งข้อมูลออกมาที่ขา SO หรืออีกแบบหนึ่งคือ จะอ่านข้อมูลจาก Main Memory Flash ออกไปที่ขา SO โดยตรงก็ได้ ไม่ต้องผ่าน Buffer 1 หรือ 2 ภายใน Chip เป็นต้น



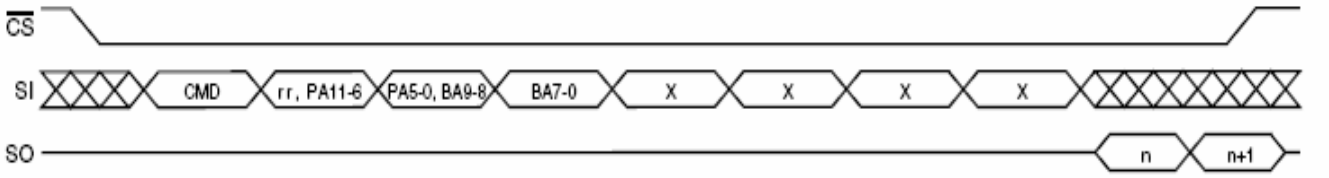
รูปที่4 แสดง บล็อกไดอะแกรม การอ่านข้อมูลจาก Chip

จากตารางที่ 1 ด้านล่างจะเป็นชุดคำสั่งอ่านข้อมูลจาก Chip หน่วยความจำ ซึ่งจะมีอยู่ด้วยกัน 5 คำสั่ง โดยมี Opcode ให้เลือกใช้ 2 ค่า ซึ่งการที่จะเลือกใช้ค่าไหนนั้นจะขึ้นอยู่กับลักษณะของ Clock SPI ที่ผู้ใช้ส่งเข้ามาทางขา SCK ว่าส่งมาในลักษณะใด โดยลักษณะของ Clock นั้นจะมีด้วยกัน 4 แบบ ซึ่งดูได้จาก Timming ที่แสดงในคำสั่งที่ใช้อ่านนั้นๆ

ตารางที่1 คำสั่งอ่าน(Read Command)

Command	SCK Mode	Opcode
Continuous Array Read	-Inactive Clock Polarity Low or High	68H
	-SPI Mode 0 or 3	E8H
Main Memory Page Read	-Inactive Clock Polarity Low or High	52H
	-SPI Mode 0 or 3	D2H
Buffer 1 Read	-Inactive Clock Polarity Low or High	54H
	-SPI Mode 0 or 3	D4H
Buffer 2 Read	-Inactive Clock Polarity Low or High	56H
	-SPI Mode 0 or 3	D6H
Status Register Read	-Inactive Clock Polarity Low or High	57H
	-SPI Mode 0 or 3	D7H

-Continuous Array Read Command : คำสั่งนี้จะเป็นการอ่านข้อมูลแบบต่อเนื่อง ซึ่งจะอ่านข้อมูลจาก Main Memory Flash ส่งออกมาที่ขา SO โดยไม่ผ่าน Buffer ภายใน เมื่ออ่านข้อมูลจนจบ Page(528Byte) ใดๆก็ตาม ในหน่วยความจำหลักแล้วอุปกรณ์ก็จะทำการอ่านข้อมูลต่อที่จุดเริ่มต้นของ Page ต่อไปและเมื่ออ่านข้อมูลมาถึงบิตสุดท้ายในหน่วยความจำหลัก อุปกรณ์ก็จะกลับไปอ่านต่อที่จุดเริ่มต้นของ Page แรก ของหน่วยความจำ โดย Opcode ของคำสั่งนี้คือ 68H หรือ E8H (ขึ้นอยู่กับลักษณะของ Clock) รูปแบบการส่ง Command จะแสดงดังรูปด้านล่าง



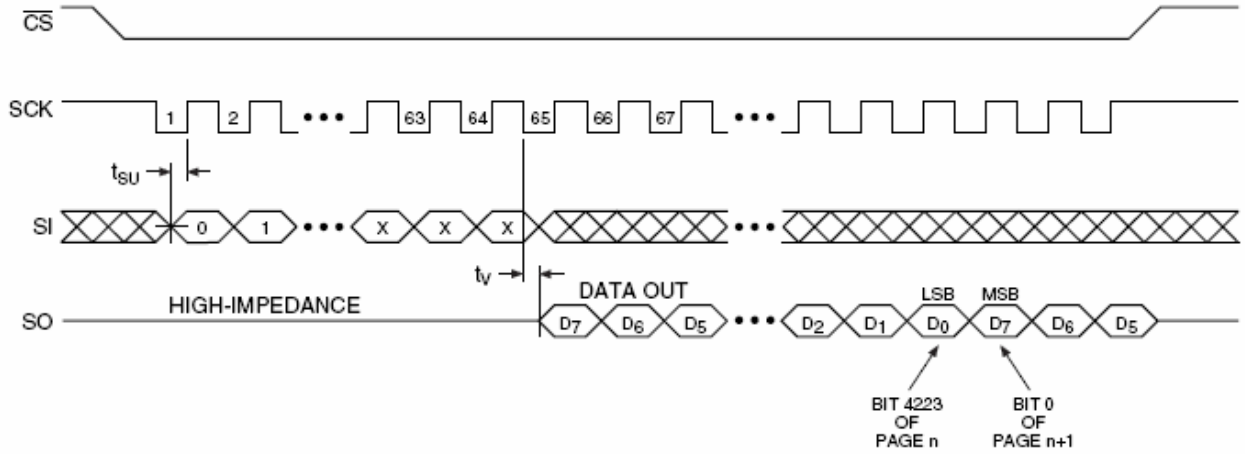
รูปที่5 แสดง การส่ง Command Continuous Array Read

จากรูปหลังจากที่ส่ง Command 4 Byte แรกตามรูปแบบที่กำหนดไว้ออกไปแล้ว จะต้องส่งข้อมูล Don't Care (0 หรือ 1 ก็ได้) ตามออกไปอีก 4 Byte ในรูปก็คือส่วนที่เป็น X จากนั้นข้อมูลที่อ่านถึงจะเริ่มถูกส่งออกมาทางขา SO ในขณะที่เริ่มต้นส่ง Opcode จนถึงกระบวนการอ่าน Data ขา $\overline{CS}$ จะต้องยังคงเป็น Low อยู่เสมอ และเมื่อมีการส่งสัญญาณจาก Low-to-High ไปยังขา $\overline{CS}$ ถึงจะเป็นการสิ้นสุดการอ่าน และขา SO ก็จะอยู่ในสภาวะ tri-state

จะเห็นว่าในคำสั่งนี้จะมี Opcode 68H และ E8H ให้เลือก 2 ชุด ซึ่งจะเลือกใช้ Opcode ใดนั้นจะขึ้นอยู่กับรูปแบบของ Clock ที่ส่งมาให้ Chip ซึ่งจะมีอยู่ด้วยกัน 4 รูปแบบ โดยดูได้จาก Timing Diagram ที่แสดงรายละเอียดการทำงานในระดับบิต ตามรูปแบบ Clock ทั้ง 4 แบบ

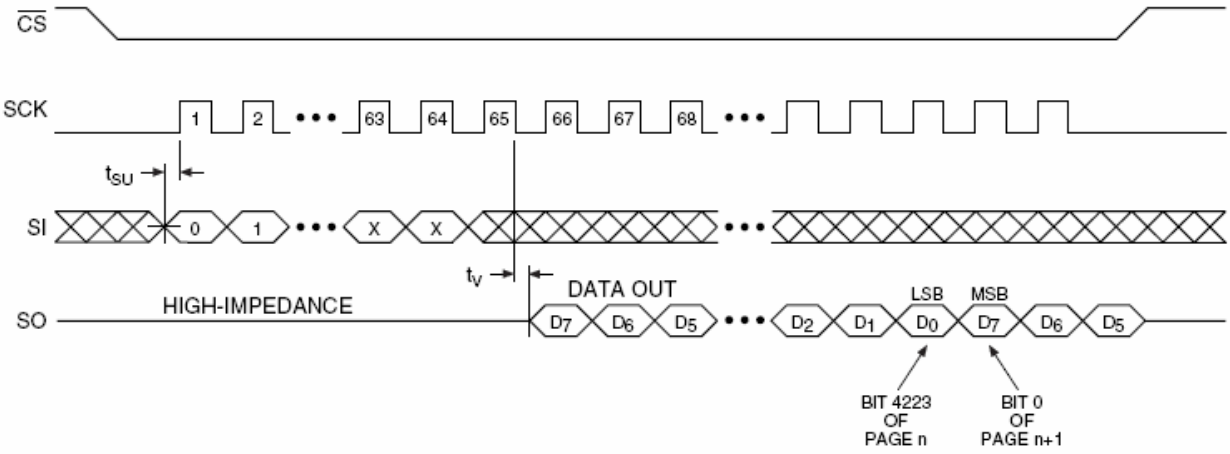
Detailed Bit-level Read Timing – Inactive Clock Polarity High

Continuous Array Read (Opcode: 68H)

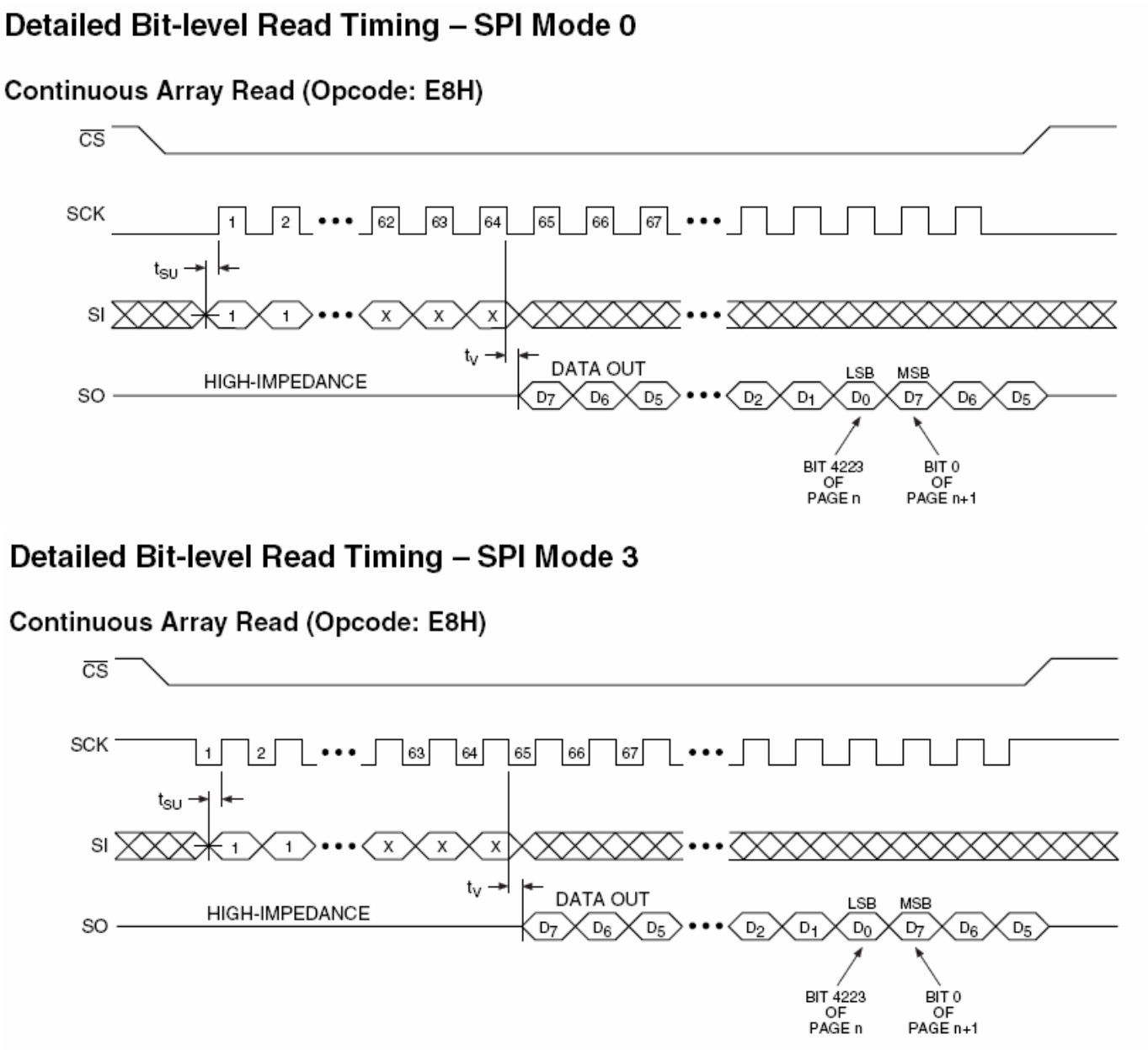


Detailed Bit-level Read Timing – Inactive Clock Polarity Low

Continuous Array Read (Opcode: 68H)

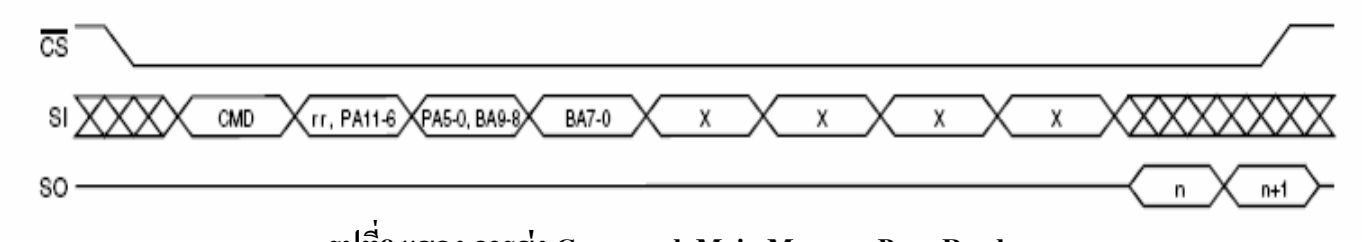


รูปที่6 แสดงการอ่านข้อมูลเมื่อใช้คำสั่ง Opcode 68H(Inactive Clock Polarity High or Low)



รูปที่7 แสดงการอ่านข้อมูลเมื่อใช้คำสั่ง Opcode E8H(SPI Clock Mode 0 or Mode 3)

-Main Memory Page Read Command : คำสั่งนี้จะให้ผู้ใช้อ่านข้อมูลโดยตรงจาก Page ใด Page หนึ่งใน 4096 Page ที่อยู่ใน Main Memory โดยไม่ผ่าน Buffer ภายใน ออกมายังขา SO เลย เริ่มต้นการอ่านนั้นจะต้องส่ง Opcode 52H หรือ D2H (ขึ้นอยู่กับรูปแบบของ Clock ที่ใช้งาน) ออกไปก่อน แล้วตามด้วย 24 บิตแอดเดรส ตามรูปแบบการส่ง Command ดังรูปที่8 สุดท้ายตามด้วยข้อมูล Don't Care (0 หรือ 1 ก็ได้) ตามออกไปอีก 4 Byte ในรูปก็คือส่วนที่เป็น X ขณะที่เริ่มต้นส่ง Opcode จนถึงกระบวนการอ่าน Data ขา $\overline{CS}$ จะต้องยังคงเป็น Low อยู่เสมอ และเมื่อมีการส่งสัญญาณจาก Low-to-High ไปยังขา $\overline{CS}$ ถึงจะเป็นการสิ้นสุดการอ่าน และขา SO ก็จะอยู่ในสภาวะ tri-state

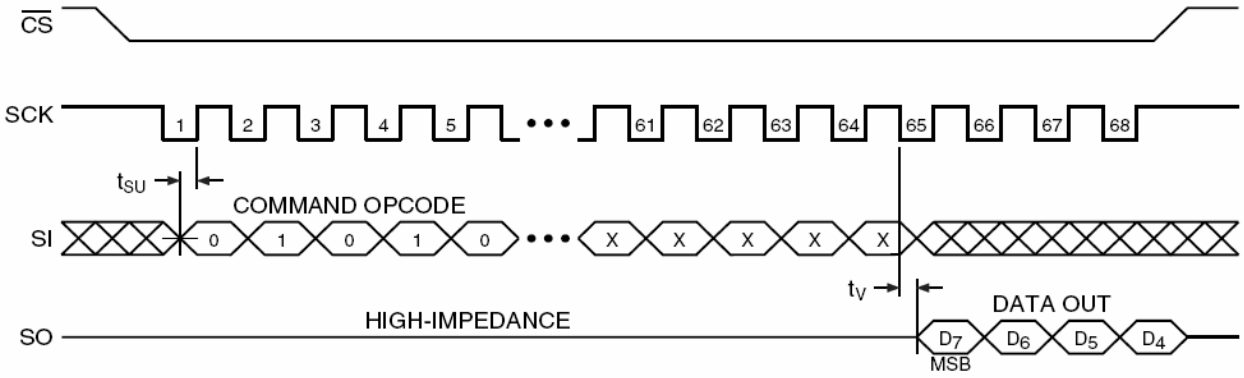


รูปที่8 แสดง การส่ง Command Main Memory Page Read

จะเห็นว่าในคำสั่งนี้จะมี Opcode ให้เลือก 2 ชุด คือ 52H และ D2H ซึ่งจะเลือกใช้ Opcode ใดนั้นจะขึ้นอยู่กับรูปแบบของ Clock ที่ส่งมาให้ Chip ซึ่งจะมีอยู่ด้วยกัน 4 รูปแบบ โดยดูได้จาก Timing Diagram ที่แสดงรายละเอียดการทำงานในระดับบิต ตามรูปแบบ Clock ทั้ง 4 แบบ

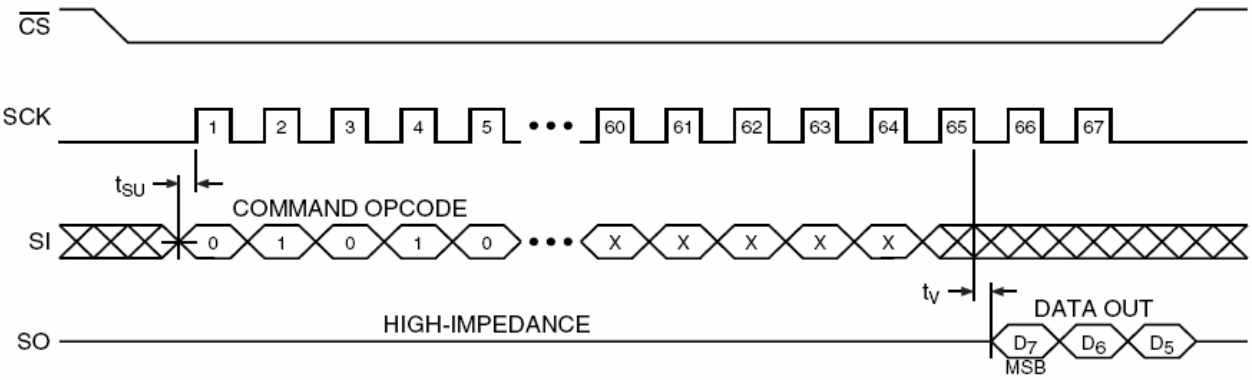
Detailed Bit-level Read Timing – Inactive Clock Polarity High

Main Memory Page Read (Opcode: 52H)

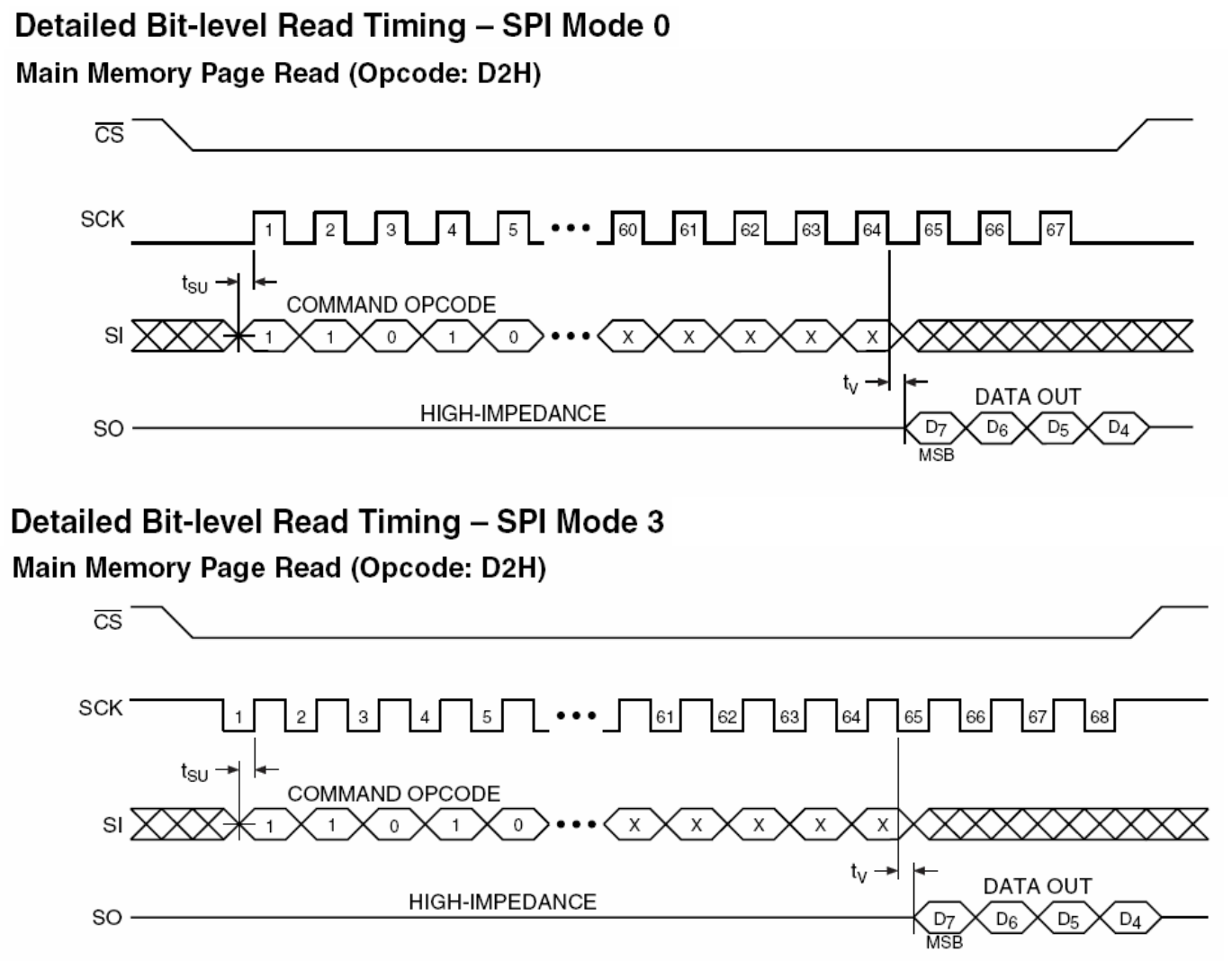


Detailed Bit-level Read Timing – Inactive Clock Polarity Low

Main Memory Page Read (Opcode: 52H)



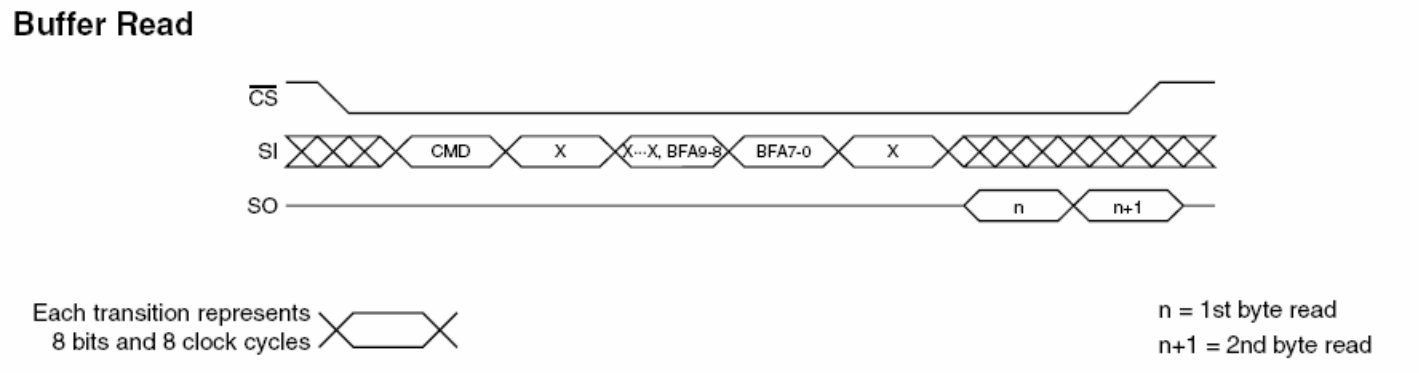
รูปที่9 แสดงการอ่านข้อมูลเมื่อใช้คำสั่ง Opcode 52H(Inactive Clock Polarity High or Low)



รูปที่10 แสดงการอ่านข้อมูลเมื่อใช้คำสั่ง Opcode D2H(SPI Clock Mode 0 or Mode 3)

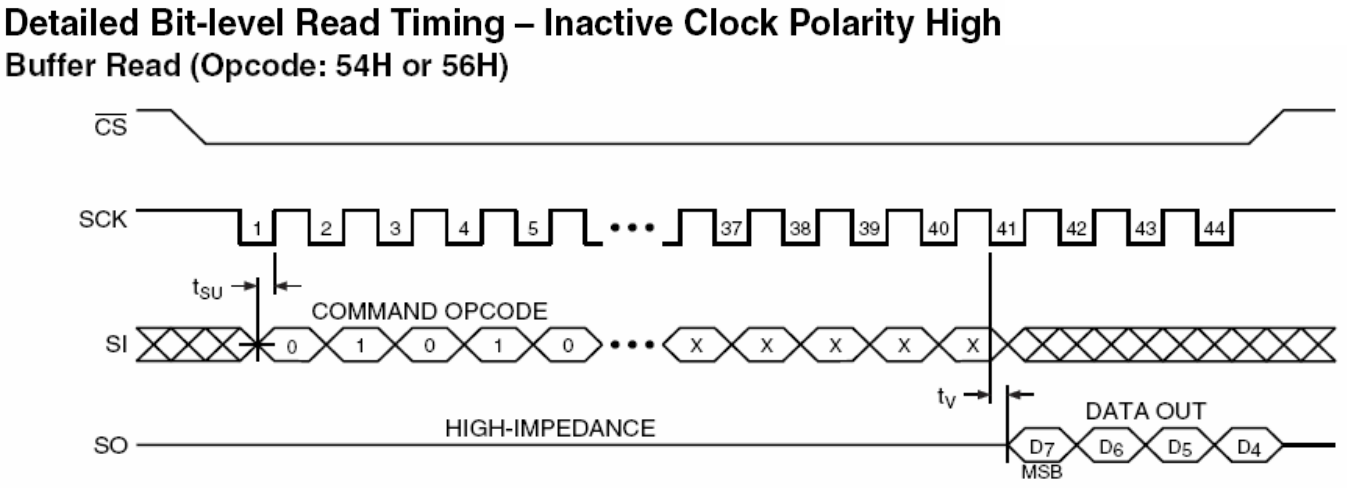
- Buffer1 Read and Buffer2 Read Command : คำสั่งนี้จะใช้สำหรับอ่านข้อมูลจาก Buffer1 หรือ Buffer2 ที่อยู่ภายใน Chip โดยถ้าใช้ Opcode 54H หรือ D4H(ขึ้นอยู่กับรูปแบบของ Clock ที่ใช้งาน) จะใช้เพื่ออ่านข้อมูลจาก Buffer1 และถ้าใช้ Opcode 56H หรือ D6H(ขึ้นอยู่กับรูปแบบของ Clock ที่ใช้งาน) จะใช้เพื่ออ่านข้อมูลจาก Buffer2

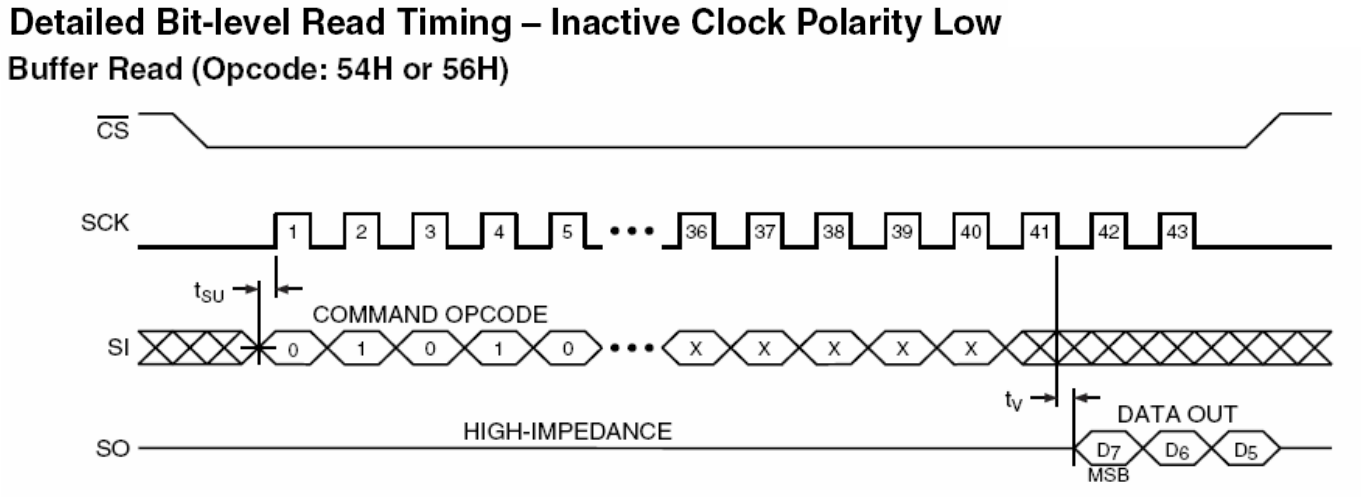
เริ่มต้นการอ่านนั้นจะต้องส่ง Opcode 8 บิต ออกไปก่อน แล้วตามด้วย 14 บิต Don't Care ตามด้วย 10 บิต แอดเดรส (BFA9-BFA0) ซึ่งเป็นตำแหน่งที่ตั้ง Byte แรกของข้อมูลที่จะอ่านจาก Buffer ซึ่ง Buffer นั้นมีขนาด 528 Byte ดังนั้นแอดเดรสจะอ้างได้ตั้งแต่ 0-527 สุดท้ายตามด้วยข้อมูล Don't Care (0 หรือ 1 ก็ได้) ตามออกไปอีก 1Byte ข้อมูลถึงจะเริ่มถูกส่งออกมาทางขา SO ดังแสดงในรูปด้านล่าง ขณะที่เริ่มต้นส่ง Opcode จนถึงกระบวนการอ่าน Data ขา $\overline{CS}$ จะต้องยังคงเป็น Low อยู่เสมอ และเมื่อมีการส่งสัญญาณจาก Low-to-High ไปยังขา $\overline{CS}$ ก็จะเป็นการสิ้นสุดการอ่าน และขา SO ก็จะอยู่ในสภาวะ tri-state



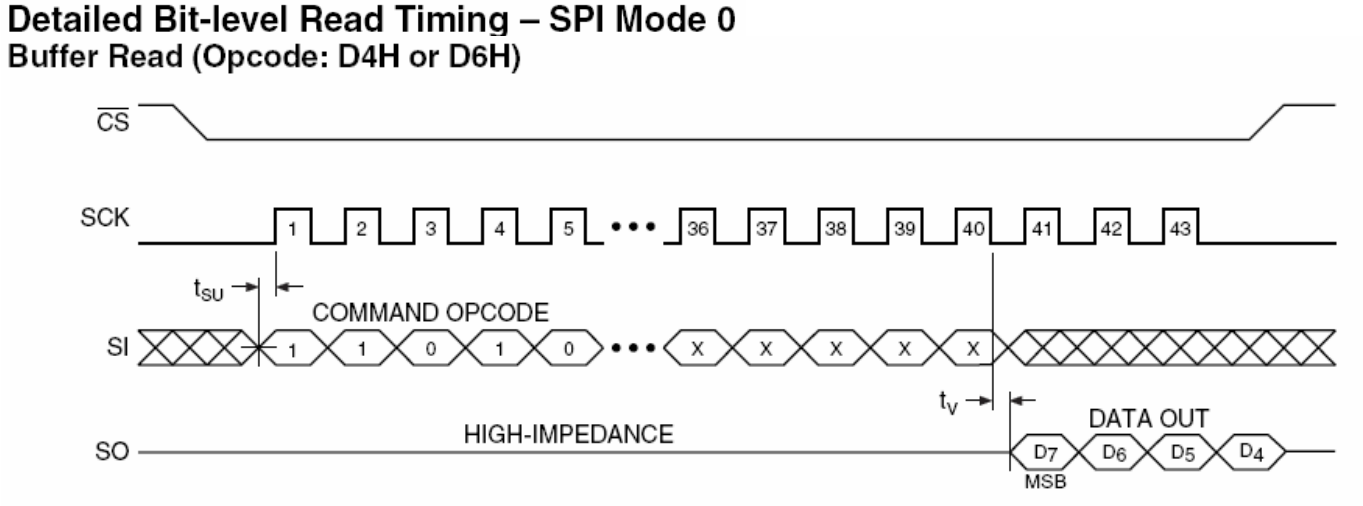
รูปที่11 แสดง การส่ง Command Buffer Read

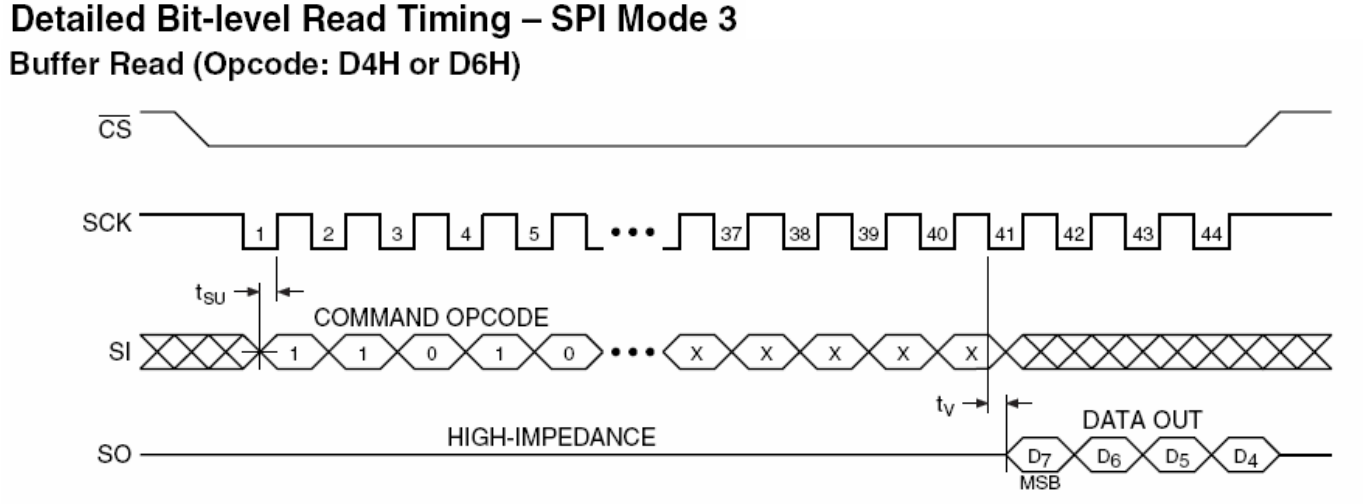
จะเห็นว่าในคำสั่งอ่าน Buffer1 และ Buffer2 นี้จะมี Opcode ให้เลือก 2 ชุด คือ 54H หรือ D4H (Buffer1) และ 56H หรือ D6H(Buffer2) ซึ่งจะเลือกใช้ Opcode ใดนั้นจะขึ้นอยู่กับรูปแบบของ Clock ที่ส่งมาให้ Chip ซึ่งจะมีอยู่ด้วยกัน 4 รูปแบบ โดยดูได้จาก Timing Diagram ที่แสดงรายละเอียดการทำงานในระดับบิต ตามรูปแบบ Clock ทั้ง 4 แบบ





รูปที่12 แสดงการอ่านข้อมูลเมื่อใช้คำสั่ง Opcode 54H or 56H (Inactive Clock Polarity High or Low)





รูปที่13 แสดงการอ่านข้อมูลเมื่อใช้คำสั่ง Opcode D4H or D6H(SPI Clock Mode 0 or Mode 3)

- Status Register Read Command : คำสั่งนี้จะใช้สำหรับอ่าน Status Register เพื่อใช้ตรวจสอบความพร้อมในการรับคำสั่งอื่นต่อไป เริ่มต้นการอ่าน Status Register นั้นจะต้องส่ง Opcode 57H หรือ D7H (ขึ้นอยู่กับรูปแบบของ Clock ที่ใช้งาน) ไปยัง Chip หลังจาก Opcod บิตสุดท้ายถูกเลื่อนเข้าไปแล้ว ข้อมูล 8 บิตของ Status register ก็จะถูกส่งออกมายังขา SO โดยบิต MSB(bit7) จะถูกส่งออกมาเป็นบิตแรก เมื่อข้อมูลถูกเลื่อนออกมาจนครบ 8 บิต การทำงานก็จะวนกลับมาเริ่มส่งค่า Status Register ในบิตที่7 จนครบ 8 บิตซ้ำอีกไปเรื่อยๆ นั่นคือ Status Register ก็จะถูกส่งออกมา Update ไปเรื่อยๆตราบที่ขา $\overline{CS}$ ยังคงเป็น Low และ SCK ยังคงทำงานอยู่

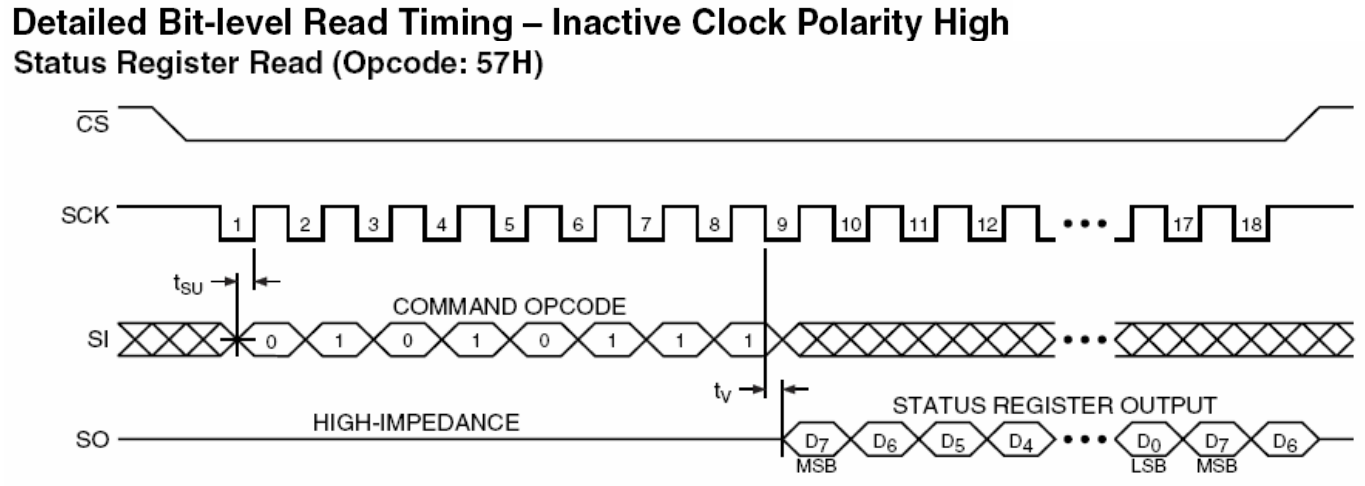
Status Register Format

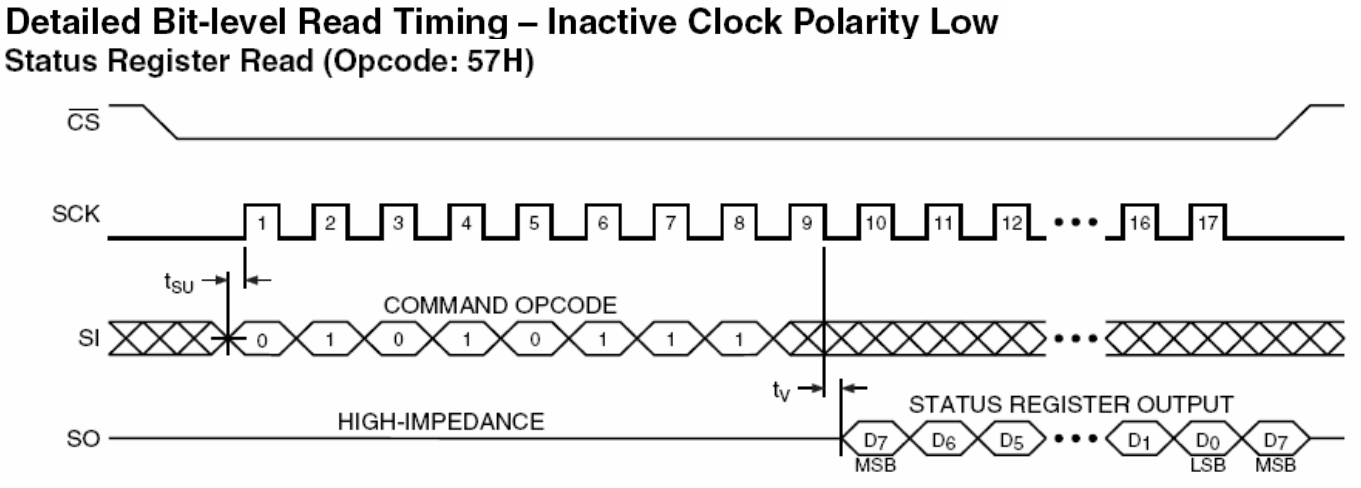
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RDY/BUSY	COMP	1	0	1	1	X	X

จากตาราง Ready/Busy Status จะแสดงในบิตที่7 ของ Status Register โดยถ้าบิต7 นี้เป็น 1 นั่นแสดงว่าอุปกรณ์พร้อม(Ready) ที่จะรับคำสั่งต่อไป แต่ถ้าบิต 7 นี้เป็น 0 แสดงว่าอุปกรณ์อยู่ในสถานะยังไม่พร้อม(Busy) ซึ่งจะมีอยู่ด้วยกัน 8 คำสั่งที่จะทำให้อุปกรณ์เกิดสภาวะ Busy ได้คือ คำสั่ง Main Memory Page to Buffer Transfer , Main Memory Page to Buffer Compare , Buffer to Main Memory Page Program with Built-in Erase , Buffer to Main Memory Page Program without Built-in Erase , Page Erase , Block Erase , Main Memory Page Program และ Auto Page Rewrite

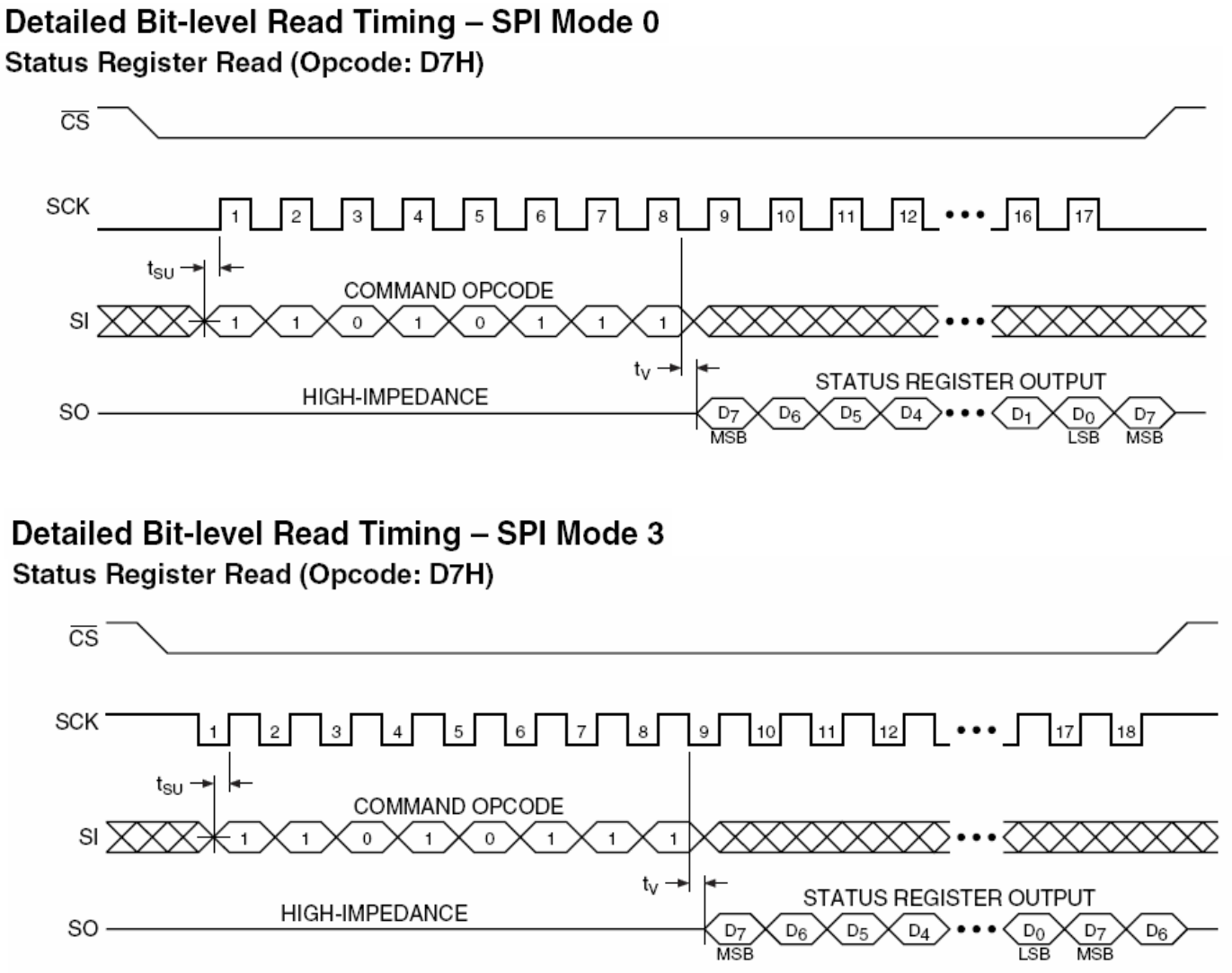
สำหรับบิตที่ 6 นั้นจะใช้แสดงสถานะการทำงานของคำสั่ง Main Memory Page to Buffer Compare โดยถ้าบิต 6 เป็น 0 แสดงว่าข้อมูลใน Memory Page เหมือนกับข้อมูลใน Buffer แต่ถ้าบิต 6 เป็น 1 แสดงว่าข้อมูลใน Main Memory Page อย่างน้อย 1 บิต ไม่เหมือนกับข้อมูลใน Buffer

จะเห็นว่าในคำสั่งอ่าน Status Register นี้จะมี Opcode ให้เลือก 2 ชุด คือ 57 หรือ D7H ซึ่งจะเลือกใช้ Opcode ใดนั้นจะขึ้นอยู่กับรูปแบบของ Clock ที่ส่งมาให้ Chip ซึ่งจะมีอยู่ด้วยกัน 4 รูปแบบ โดยดูได้จาก Timing Diagram ที่แสดงรายละเอียดการทำงานในระดับบิต ตามรูปแบบ Clock ทั้ง 4 แบบ





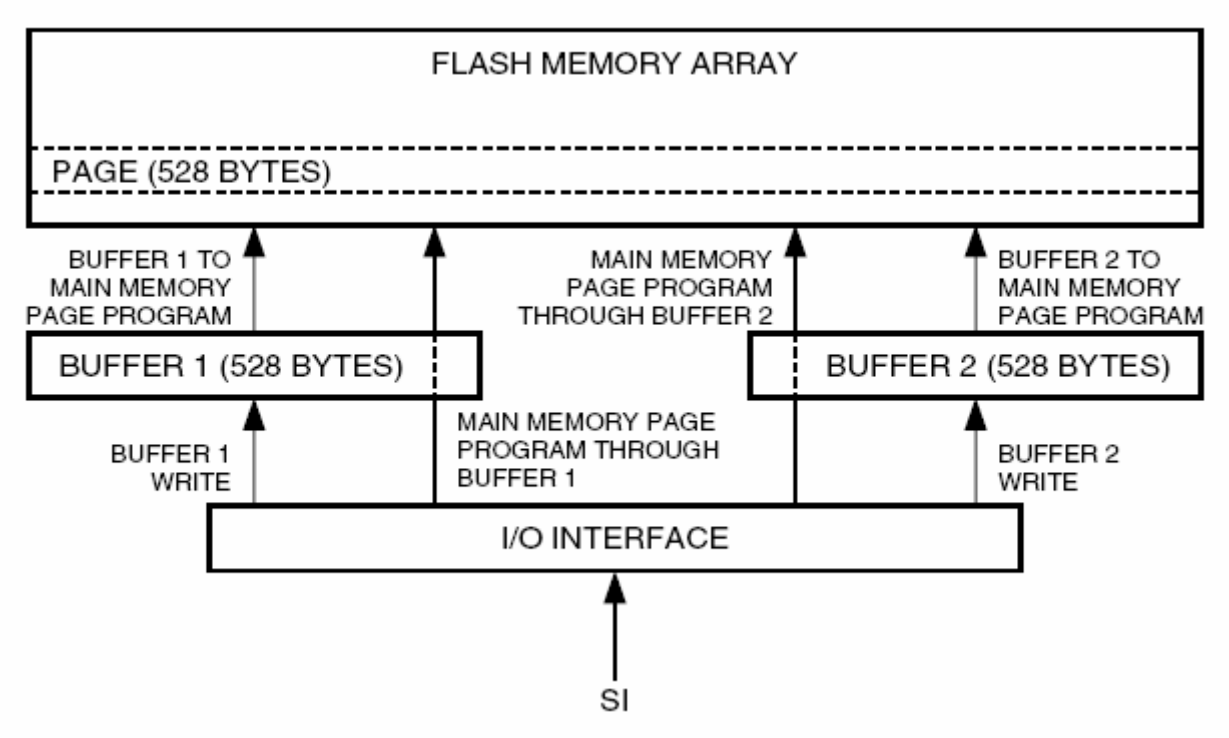
รูปที่14 แสดงการอ่าน Status Register เมื่อใช้คำสั่ง Opcode 57H (Inactive Clock Polarity High or Low)



รูปที่15 แสดงการอ่าน Status Register เมื่อใช้คำสั่ง Opcode D7H (SPI Clock Mode 0 or Mode 3)

การเขียนและคำสั่งที่ใช้ในการเขียนข้อมูล

การเขียนข้อมูลลงใน Chip นั้นจะแสดงให้เห็นดังบล็อกไดอะแกรมด้านล่าง ซึ่งจะสอดคล้องกับคำสั่งที่ใช้ในการเขียนดังตารางที่2 จากบล็อกไดอะแกรมจะเห็นว่าการเขียนข้อมูลนั้นมีการเขียนได้หลายแบบ ซึ่งแต่ละแบบคำสั่งที่ใช้ในการเขียนก็จะแตกต่างกันไป ผู้ใช้จึงต้องทำความเข้าใจในแต่ละคำสั่งเสียก่อนเพื่อจะเลือกใช้คำสั่งได้ถูกต้อง จะสังเกตเห็นว่าคำสั่งที่ใช้สำหรับเขียน ในแต่ละคำสั่งนั้น จะมี Opcode เพียงชุดเดียว ดังนั้นจะเลือกใช้รูปแบบของสัญญาณ Clock แบบใดแบบหนึ่งใน 4 แบบที่ได้กล่าวไปข้างต้นในคำสั่งอ่านก็ได้ แต่ขอแนะนำให้ใช้รูปแบบของการส่งสัญญาณ Clock แบบเดียวกับที่ใช้กับคำสั่งอ่าน



รูปที่16 แสดง บล็อกไดอะแกรม การเขียนข้อมูลลง Chip

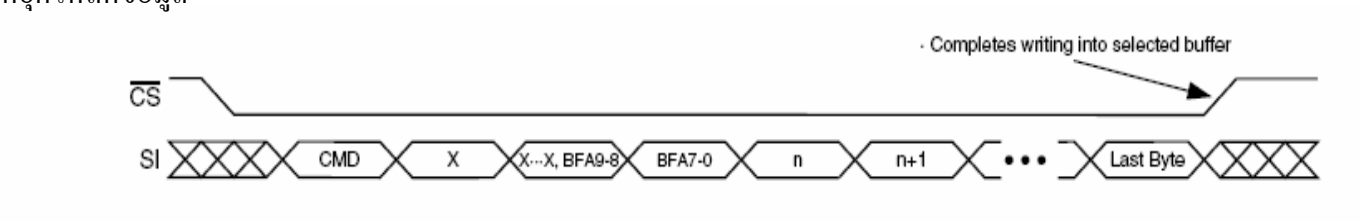
จากตารางที่ 2 ด้านล่างจะเป็นชุดคำสั่งเขียนข้อมูลลง Chip และลบข้อมูลออกจาก Chip หน่วยความจำ ซึ่งจะมีอยู่ด้วยกัน 10 คำสั่ง ซึ่งรูปแบบการส่ง Command ก็จะเหมือนกับในรูปที่ 3

ตารางที่2 คำสั่งเขียน(Write Command)

Command	SCK Mode	Opcode
Buffer 1 Write	Any	84H
Buffer 2 Write	Any	87H
Buffer1 to Main Memory Page Program with Built-in Erase	Any	83H
Buffer2 to Main Memory Page Program with Built-in Erase	Any	86H
Buffer1 to Main Memory Page Program without Built-in Erase	Any	88H
Buffer2 to Main Memory Page Program without Built-in Erase	Any	89H
Page Erase	Any	81H
Block Erase	Any	50H
Main Memory Page Program through Buffer1	Any	82H
Main Memory Page Program through Buffer2	Any	85H

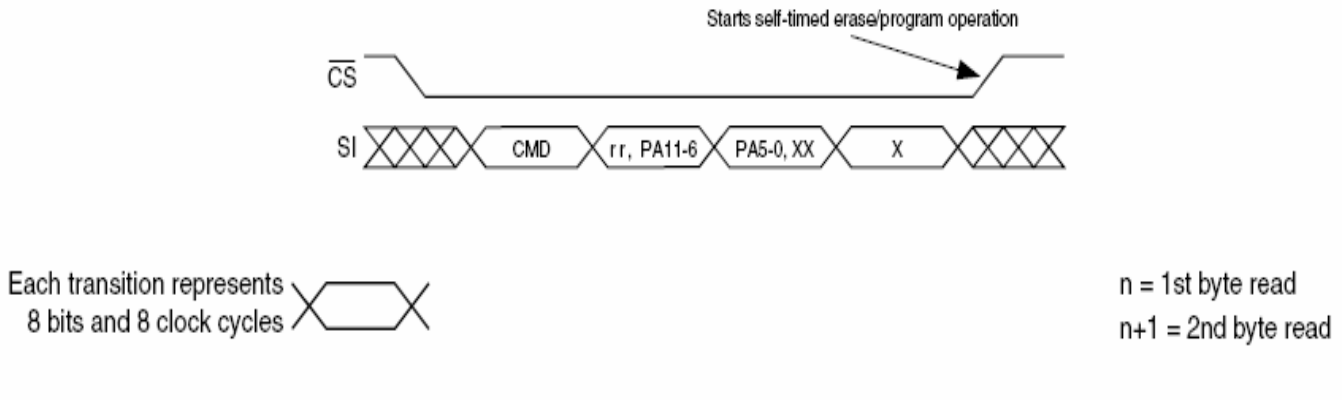
Any = เลือกใช้ Clock แบบใดแบบหนึ่งใน 4 แบบ ได้แก่ Inactive Clock Polarity Low , Inactive Clock Polarity Low , SPI Mode 0 และ SPI Mode 3

- Buffer 1 Write และ Buffer 2 Write : 2 คำสั่งนี้จะใช้สำหรับส่งข้อมูลจากภายนอกผ่านขา SI เข้าไปเก็บไว้ภายใน Buffer1 หรือ Buffer2 ของตัว Chip ซึ่งการโหลดข้อมูลเข้าไปยัง Buffer นั้น จะต้องส่ง Opcode 8 บิต คือ 84H สำหรับ Buffer1 หรือ 87H สำหรับ Buffer2 แล้วตามด้วย 14 บิต Don't Care และ 10 บิตแอดเดรส(BFA9-BFA0) ตามลำดับ ซึ่ง 10 บิตแอดเดรสนี้ก็คือ ตำแหน่งแอดเดรสเริ่มต้นใน Buffer ที่จะทำการเขียน สุดท้ายก็ตามด้วยข้อมูลที่จะเขียน 528 Byte ข้อมูลจะถูกโหลดเข้าไปยัง Buffer จนกระทั่ง ขา $\overline{CS}$ เปลี่ยนสถานะจาก Low เป็น High ถึงจะเป็นการหยุดโหลดข้อมูล



รูปที่17 แสดง การส่ง Command Buffer Write

- Buffer1 หรือ Buffer2 to Main Memory Page Program With Built-In Erase : 2 คำสั่งนี้จะใช้สำหรับเคลื่อนย้ายข้อมูล จาก Buffer 1 หรือ Buffer 2 เข้าไปเก็บไว้ยัง Main Memory ในการเริ่มต้นการทำงานจะต้องส่ง 8 บิต opcode 83H สำหรับ Buffer1 หรือ 86H สำหรับ Buffer2 แล้วตามด้วย 2 บิต สงวน(rr) และ 12 บิตแอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Page ใน Main Memory ที่จะทำการเขียนข้อมูลเข้าไป สุดท้ายตามด้วย 10 บิต Don't Care โดยจะเป็นไปตามรูปที่18 หลังจากส่ง Command ครบ 4 Byte แล้ว และทำให้ขา $\overline{CS}$ เปลี่ยนระดับจาก Low เป็น High จะทำให้ Page ที่ถูกเลือกใน Main Memory ถูกลบข้อมูลออกทั้งหมด จากนั้นก็จะเริ่มทำการย้ายข้อมูลจาก Buffer ไปเก็บไว้ยัง Page ใน Main Memory ที่ได้เลือกไว้ ระหว่างที่มีการเคลื่อนย้ายข้อมูลนี้ Status Register จะแสดงสภาวะ Busy



รูปที่18 Timming แสดง การส่ง Command Buffer to Main Memory Page Program

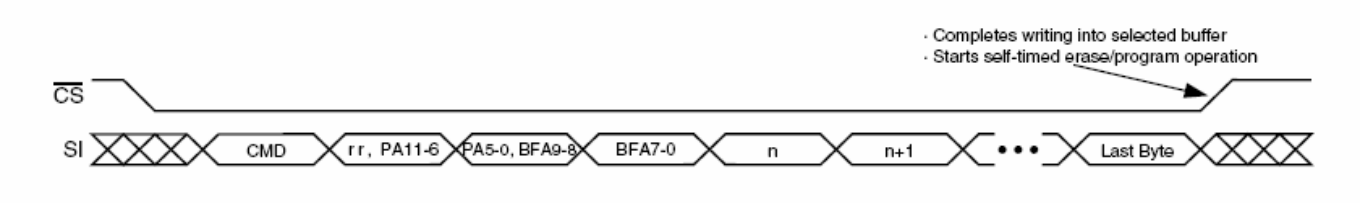
- Buffer1 หรือ Buffer2 to Main Memory Page Program Without Built-In Erase : 2 คำสั่งนี้จะใช้สำหรับเคลื่อนย้ายข้อมูล จาก Buffer 1 หรือ Buffer 2 เข้าไปเก็บไว้ยัง Main Memory เช่นกัน แต่จะไม่มีการลบข้อมูล ที่อยู่ใน Page ออกก่อน ในการเริ่มต้นการทำงานจะต้องส่ง 8 บิต Opcode 88H สำหรับ Buffer1 หรือ 89H สำหรับ Buffer2 แล้วตามด้วย 2 บิต สงวน(rr) และ 12 บิตแอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Page ใน Main Memory ที่จะทำการเขียนข้อมูลเข้าไป สุดท้ายตามด้วย 10 บิต Don't Care หลังจากส่ง Command ครบ 4 Byte แล้ว และทำให้ขา $\overline{CS}$ เปลี่ยนระดับจาก Low เป็น High ก็จะเริ่มทำการย้ายข้อมูลจาก Buffer ไปเก็บไว้ยัง Page ใน Main Memory ที่ได้เลือกไว้ ระหว่างที่มีการเคลื่อนย้ายข้อมูลนี้ Status Register จะแสดงสภาวะ Busy

- Page Erase : คำสั่งนี้จะถูกใช้สำหรับลบข้อมูลใน Page ของ Main Memory ซึ่งจะเป็นประโยชน์ต่อการใช้คำสั่ง Buffer to Main Memory Page Program Without Built-In Erase ตามออกมาหลังคำสั่ง Page Erase ในการเริ่มต้นการทำงานจะต้องส่ง 8บิต Opcode 81H แล้วตามด้วย 2 บิต สงวน(rr) และ 12 บิตแอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Page ใน Main Memory ที่จะทำการลบข้อมูลออก สุดท้ายตามด้วย 10 บิต Don't Care หลังจากส่ง Command ครบ 4 Byte แล้ว และทำให้ขา $\overline{CS}$ เปลี่ยนระดับจาก Low เป็น High ก็จะเริ่มทำการลบข้อมูลใน Page ที่ถูกเลือกออก ระหว่างที่มีการลบข้อมูลออกนี้ Status Register จะแสดงสภาวะ Busy

- Block Erase : คำสั่งนี้จะใช้สำหรับลบข้อมูลเป็น Block ซึ่งก็คือการลบข้อมูล ใน Page ของ Main Memory ออกครั้งละ 8 Page ซึ่งจะเป็นประโยชน์ต่อการใช้คำสั่ง Buffer to Main Memory Page Program Without Built-In Erase เพื่อลดเวลาในการเขียนข้อมูลจำนวนมากลงไปยัง Main memory ในการเริ่มต้นการทำงานจะต้องส่ง 8บิต Opcode 50H แล้วตามด้วย 2 บิต สงวน(rr) และ 9 บิตแอดเดรส (PA11-PA3) ซึ่งเป็นตำแหน่ง Block ของ 8 Pageใน Main Memory ที่จะทำการลบข้อมูลออก สุดท้ายตามด้วย 13 บิต Don't Care หลังจากส่ง Command ครบ 4 Byte แล้ว และทำให้ขา $\overline{CS}$ เปลี่ยนระดับจาก Low เป็น High ก็จะเริ่มทำการลบข้อมูลใน Block ที่ถูกเลือกออก ระหว่างที่มีการลบข้อมูลออกนี้ Status Register จะแสดงสภาวะ Busy

ในการส่ง Command (4Byte)ออกไปแต่ละครั้งนั้นผู้ใช้จะต้องทำให้ขา $\overline{CS}$ เปลี่ยนสถานะจาก High เป็น Low เสมอ และขา $\overline{CS}$ จะเปลี่ยนสถานะจาก Low มาเป็น High อีกครั้งเมื่อไหร่นั้นจะขึ้นอยู่กับรูปแบบของคำสั่งแต่ละคำสั่งที่ใช้

- Main Memory Page Program through Buffer1 หรือ Buffer2 : 2 คำสั่งนี้จะเป็นการรวมเอาการทำงานของคำสั่ง Buffer Write และ Buffer to Main Memory Page Program with Built-in Erase เข้าไว้ด้วยกัน นั่นคือเมื่อใช้คำสั่งนี้ ข้อมูลจะถูกเลื่อนจากขา SI เข้าไปยัง Buffer1 หรือ Buffer2 จากนั้นข้อมูลก็จะถูกเลื่อนต่อเข้าไปยัง Main Memory ให้โดยอัตโนมัติ ในการเริ่มต้นการทำงานจะต้องส่ง 8บิต Opcode 82H สำหรับ Buffer1 และ 85H สำหรับ Buffer2 แล้วตามด้วย 2 บิต สงวน(rr) และ 12 บิตแอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Pageใน Main Memory ที่จะทำการเขียนข้อมูลเข้าไป สุดท้ายตามด้วย 10 บิต แอดเดรส(BFA9-BFA0) ซึ่งจะเป็นตำแหน่งแอดเดรสเริ่มต้น ของ Buffer ที่จะใช้ทำการเขียนข้อมูลเข้าไป หลังจากส่ง Command ครบ 4 Byte แล้ว จะต้องทำการส่งข้อมูลตามไป ข้อมูลก็จะถูกเลื่อนเข้าทางขา SI และถูกเก็บไว้ยัง Buffer ในขณะที่เลื่อนข้อมูลเข้าไปนี้ ขา $\overline{CS}$ จะต้องยังคงเป็น Low อยู่ หลังจากโหลดข้อมูลครบ 528 byte แล้ว และทำให้ขา $\overline{CS}$ เปลี่ยนระดับจาก Low เป็น High ข้อมูลเดิมใน Page ที่ถูกเลือกไว้ก็จะถูกลบออกไป จากนั้นข้อมูลที่ถูกเก็บไว้ใน Buffer ที่โหลดเข้ามาในตอนแรกก็จะถูกเคลื่อนย้ายเข้าไปเก็บไว้ใน Page ของ Main Memory โดยอัตโนมัติ ในระหว่างการทำงานนี้ Status Register จะแสดงสภาวะ Busy



รูปที่19 Timming แสดง การส่ง Command Main Memory Page Program through Buffer

คำสั่งเพิ่มเติม

สำหรับคำสั่งเพิ่มเติมนี้จะแสดงดังในตารางที่ 3 ซึ่งจะเป็นคำสั่งในการเคลื่อนย้ายข้อมูลภายใน Chip ระหว่าง Buffer ไปยัง Main Memory หรือ จาก Main Memmory ไปยัง Buffer โดยรูปแบบ Clock SPI ที่จะใช้งานในคำสั่งชุดนี้จะเป็นรูปแบบใดรูปแบบหนึ่งใน 4 แบบ ที่กล่าวไปใน Mode ของคำสั่งอ่าน ก็ได้

ตารางที่3 คำสั่งเพิ่มเติม (Additional Command)

Command	SCK Mode	Opcode
Main Memory Page to Buffer 1 Transfer	Any	53H
Main Memory Page to Buffer 2 Transfer	Any	55H
Main Memory Page to Buffer 1 Compare	Any	60H
Main Memory Page to Buffer 2 Compare	Any	61H
Auto Page Rewrite through Buffer1	Any	58H
Auto Page Rewrite through Buffer1	Any	59H

Any = เลือกใช้ Clock แบบใดแบบหนึ่งใน 4 แบบ ได้แก่ Inactive Clock Polarity Low , Inactive Clock Polarity Low , SPI Mode 0 และ SPI Mode 3

- Main Memory Page to Buffer 1 Transfer และ Main Memory Page to Buffer 2 Transfer : 2 คำสั่งนี้จะเป็นการเคลื่อนย้ายข้อมูลครั้งละ 1 Page(528 byte)จาก Main Memory ไปเก็บไว้ที่ Buffer1 หรือ Buffer2 การทำงานจะเริ่มต้นด้วยการ ส่ง 8 บิต Opcode 53H สำหรับ Buffer1 และ 55H สำหรับ Buffer2 ตามด้วย 2 บิตสงวน(rr) และ 12 บิตแอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Page ใน Main Memory ที่จะทำการอ่าน สุดท้ายตามด้วย 10 บิต Don't Care ซึ่งจะเป็นไปตามรูปแบบการส่ง Command ในรูปที่ 3 ในขณะที่เริ่มทำการส่ง Opcode ออกไปที่ขา SI จนครบ 4 Byte ขา $\overline{CS}$ จะต้องเป็น Low และข้อมูลจะเริ่มถูกเคลื่อนย้ายจาก Main Memory ไปยัง Buffer ก็ต่อเมื่อ ขา $\overline{CS}$ ถูกเปลี่ยนสถานะจาก Low เป็น High ในระหว่างที่มีการเคลื่อนย้ายข้อมูลนี้ สามารถจะอ่าน Status Register ได้ว่าการเคลื่อนย้ายข้อมูลสมบูรณ์หรือยัง

- Main Memory Page to Buffer 1 Compare และ Main Memory Page to Buffer 2 Compare : 2 คำสั่งนี้จะเป็นการเปรียบเทียบข้อมูลใน Page ของ Main Memory กับข้อมูลที่อยู่ใน Buffer1 หรือ Buffer2 การทำงานจะเริ่มต้นด้วยการ ส่ง 8 บิต Opcode 60H สำหรับ Buffer1 และ 61H สำหรับ Buffer2 ตามด้วย 2 บิตสงวน(rr) และ 12 บิต แอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Page ใน Main Memory ที่จะใช้ทำการ Compare ข้อมูล กับ Buffer สุดท้ายตามด้วย 10 บิต Don't Care ซึ่งจะเป็นไปตามรูปแบบการส่ง Command ในรูปที่ 3 ในขณะที่เริ่มทำการส่ง Opcode ออกไปที่ขา SI จนครบ 4 Byte ขา $\overline{CS}$ จะต้องเป็น Low และเมื่อขา $\overline{CS}$ เปลี่ยนระดับสัญญาณจาก Low เป็น High ข้อมูล 528 Byte ใน Main Memory Page ที่ถูกเลือก ก็จะถูกเปรียบเทียบกับข้อมูล 528 byte ที่อยู่ใน Buffer 1 หรือ Buffer 2 ในระหว่างที่ทำการเปรียบเทียบข้อมูลอยู่นี้ Status Register จะแสดงสภาวะ Busy และเมื่อการ Compare สมบูรณ์ ผลของการ Compare ก็จะถูก Update ในบิตที่6 ของ Register Status

- Auto Page Rewrite through Buffer1 หรือ Buffer 2 : 2 คำสั่งนี้จะเป็นการรวมเอาการทำงานของคำสั่ง Main Memory Page to Buffer Transfer และ Buffer to Main Memory Page Program with Built-in Erase เข้าไว้ด้วยกัน โดยอันดับแรกข้อมูลใน Page จะถูกส่งออกจาก Main Memory ไปยัง Buffer 1 หรือ Buffer 2 จากนั้นข้อมูลเดียวกันนี้จาก Buffer 1 หรือ Buffer 2 ก็จะถูกส่งกลับเข้าไป ยัง Page เดิมของ Main Memory อีกครั้งหนึ่ง การใช้งานของคำสั่งจะเริ่มต้นด้วยการ ส่ง 8 บิต Opcode 58H สำหรับ Buffer1 หรือ 59H สำหรับ Buffer2 ตามด้วย 2 บิตสงวน(rr) และ 12 บิต แอดเดรส (PA11-PA0) ซึ่งเป็นตำแหน่ง Page ใน Main Memory ที่จะใช้ทำการ Rewrite ข้อมูล สุดท้ายตามด้วย 10 บิต Don't Care ซึ่งจะเป็นไปตามรูปแบบการส่ง Command ในรูปที่ 3 ในขณะที่เริ่มทำการส่ง Opcode ออกไปที่ขา SI จนครบ 4 Byte ขา $\overline{CS}$ จะต้องเป็น Low และเมื่อขา $\overline{CS}$ เปลี่ยนระดับสัญญาณจาก Low เป็น High ข้อมูลใน Page ก็จะถูกส่งจาก Main Memory ไปยัง Buffer และข้อมูลเดียวกันนี้ที่อยู่ใน Buffer ก็จะถูกส่งกลับเข้ามายัง Page เดิมของ Main Memory อีกครั้ง ในระหว่างที่ทำงานนี้ Status Register จะแสดงสภาวะ Busy

ตารางที่4 Detailed Bit-level Addressing Sequence

Opcode	Address Byte								Address Byte								Address Byte								Additional Don't Care Byte Required
	Reserved	Reserved	PA11	PA10	PA9	PA8	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	
50H	r	r	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
52H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	4 Byte
53H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
54H	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	1 Byte
55H	r	r	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
56H	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	1 Byte
57H	N/A								N/A								N/A								N/A
58H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A

ตารางที่4 Detailed Bit-level Addressing Sequence(ต่อ)

Opcode	Address Byte								Address Byte								Address Byte								Additional Don't Care Byte Required
	Reserved	Reserved	PA11	PA10	PA9	PA8	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	
59H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
60H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
61H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
68H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	4 Byte
81H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
82H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	N/A
83H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
84H	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	N/A
85H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	N/A
86H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
87H	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	N/A
88H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
89H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
D2H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	4 Byte
D4H	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	1 Byte
D6H	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	1 Byte
D7H	N/A								N/A								N/A								N/A
E8H	r	r	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	4 Byte

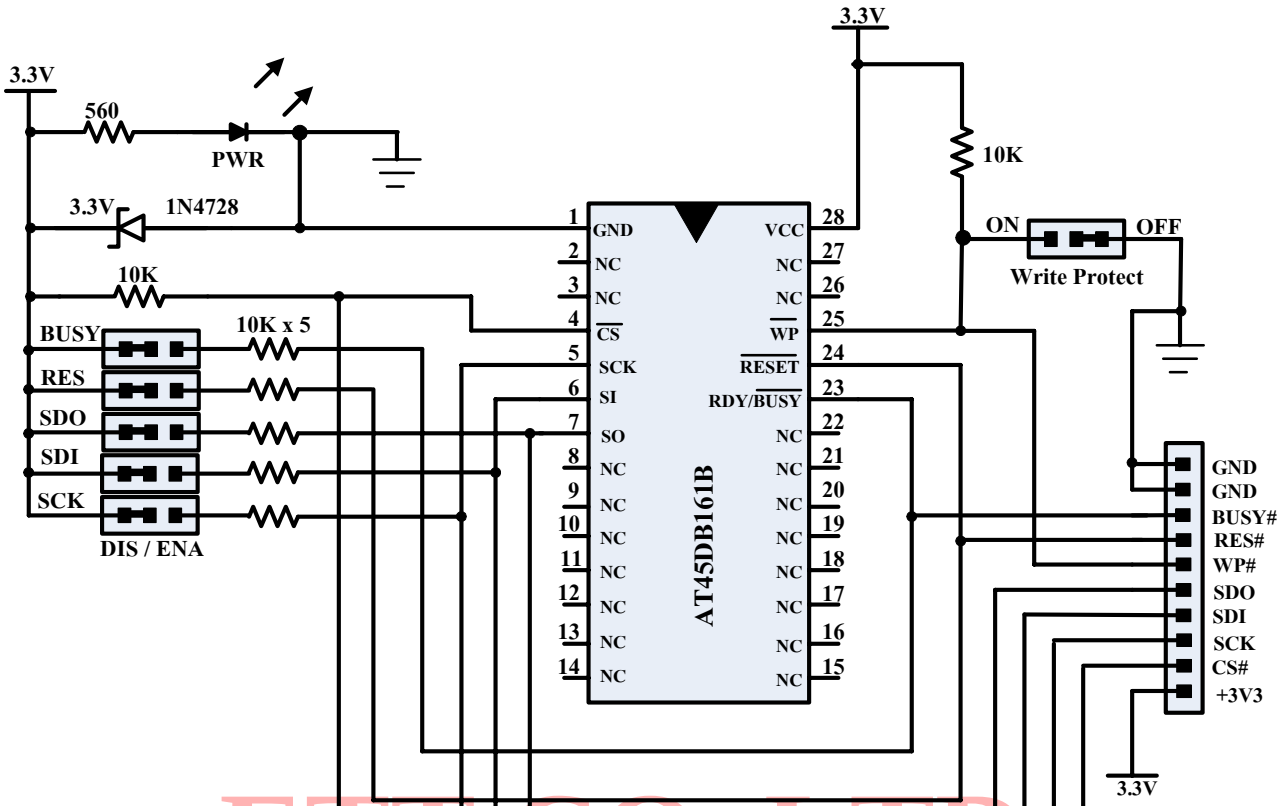
Note: r = Reserved bit (r = 0)

P = Page Address Bit (Page = 0-4095)

B = Byte/Buffer Address Bit (Address = 0-527)

x = Don't Care

สำหรับตารางที่ 4 นี้จะเป็นการแสดงรายละเอียดในระดับบิตของการส่งชุด Command ในแต่ละ Command ซึ่งลำดับการส่ง Command นั้นสามารถอ้างอิงได้จากรูปที่ 3 ซึ่งจากตารางเริ่มต้นจะเป็นการส่ง Opcode ของคำสั่ง 1 Byte แล้วตามด้วย Address Byte อีก 3 Byte ซึ่ง Address Byte ทั้ง 3 Byte นี้ก็จะประกอบไปด้วยบิตสงวน(r) 2 บิต ,ตำแหน่ง Page ใน Main Memory (PA0-PA11) 12 บิต โดยสามารถอ้างอิงค่าได้ตั้งแต่ 0-4095 , ตำแหน่ง Address ของ Page หรือ ของ Buffer (BA0-BA9) 10 บิต โดยสามารถอ้างอิงค่าได้ตั้งแต่ 0-527 และสุดท้ายคือจำนวน Byte Don't Care ที่ต้องส่งตามออกไปในบางคำสั่ง ซึ่งจะเป็น 0 หรือ 1 ก็ได้



รูป วงจร ET-MINI SPI DATA FLASH