อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

Single chip VGA Controller

โดย

นายปริญญา โพธิ์คำ รหัส 47010435

นายพลวัฒน์ กลับคง รหัส 47010493

อาจารย์ที่ปรึกษา

ดร.กสิน วิเชียรชม

ปริญญานิพนธ์สำหรับปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

Single chip VGA Controller

โดย

นายปริญญา โพธิ์คำ

นายพลวัฒน์ กลับคง

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็กทรอนิกส์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

ปริญญานิพนธ์ ปีการศึกษา 2550

ภาควิชา อิเล็กทรอนิกส์

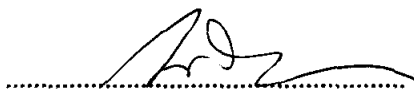
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

(Single chip VGA Controller)

ผู้จัดทำ 1.นายปริญญา โพธิ์คำ รหัส 47010435 ชั้นปีที่ 4

2 .นายพลวัฒน์ กลับคง รหัส 47010493 ชั้นปีที่ 4



.....อาจารย์ที่ปรึกษา

(ดร.กสิน วิเชียรชม)

อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

นายปริญญา โพธิ์คำ รหัส 47010435
นายพลวัฒน์ กลับคง รหัส 47010493
ดร.กสิน วิเชียรชม อาจารย์ที่ปรึกษา
ปีการศึกษา 2550

บทคัดย่อ

เครื่องแสดงภาพบนจอมอนิเตอร์ผ่านทางพอร์ต D-Sub ทำหน้าที่แสดงไฟล์ภาพที่มีอยู่ใน SD Card ออกทางจอมอนิเตอร์ ซึ่งในอุปกรณ์ชิ้นนี้มีส่วนการทำงาน 2 ส่วน คือ ส่วนเก็บข้อมูล โดยจะนำข้อมูลจาก SD Card ซึ่งจะมีการเก็บข้อมูลภาพในรูปแบบของ bitmap โดยใช้ระบบไฟล์แบบ FAT32 และส่วนควบคุมการทำงานซึ่งใช้ไมโครคอนโทรเลอร์อ่านไฟล์จาก SD Card นำมาประมวลผล ส่งเป็นสัญญาณภาพผ่านพอร์ต D-Sub เพื่อแสดงผลบนจอภาพภายนอก การติดต่อระหว่างไมโครคอนโทลเลอร์กับ SD Card ใช้การเชื่อมต่อแบบ SPI โดยอุปกรณ์ชิ้นนี้สามารถที่จะควบคุมการเลือกดูไฟล์ภาพได้

Single chip VGA Controller

Mr. Parinya Phokham ID.47010435
Mr. Phonlawat Klubkong ID.47010493
Dr. Kasin Vichienchom Advisor
Education Year 2007

Abstract

This report describes a design of the device that generates VGA signal to display image data from the SD Card. It consists of two parts, a data storage unit and a VGA signal generator. The SD Card stores the image file in Bitmap using FAT32 file system and interfaces with the generator unit, the ARM7 microcontroller, via SPI Bus. The ARM7 generates VGA signal to drive the external monitor through D-Sub port.

กิตติกรรมประกาศ

โครงงานอุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์ ซึ่งประกอบด้วยด้วยชิ้นงานและเอกสารประกอบโครงงานนี้ จะสำเร็จลุล่วงมาด้วยดีมิได้หากขาด อาจารย์กสิน วิเชียรชม อาจารย์ที่ปรึกษาผู้คอยให้คำแนะนำ และดูแลอย่างใกล้ชิดมาโดยตลอด พร้อมทั้งการให้ความช่วยเหลือจากเพื่อนในภาควิชา สุดท้ายบุคคลที่จะลืมมิได้เลยคือ ผู้ปกครองซึ่งคอยสนับสนุนและคอยให้กำลังใจเสมอมา

นายปริญญา โพธิ์คำ
นายพลวัฒน์ กลับคง
ผู้จัดทำ

สารบัญ

สารบัญรูป

หน้า

สารบัญรูป

หน้า

สารบัญตาราง

หน้า

บทที่ 1

บทนำ

1.1 ความเป็นมาของโครงงาน

ปัจจุบันเทคโนโลยีในการเก็บรูปภาพเป็นไฟล์ดิจิตอล มีการใช้งานแพร่หลายมากขึ้นโดยวิธีการนำภาพออกมาแสดงนั้นมีหลายรูปแบบ ทั้งเปิดดูในคอมพิวเตอร์ เครื่องเล่นไฟล์ต่าง ๆ รวมทั้งพิมพ์ออกมาเป็นรูปภาพ โดยผู้จัดทำต้องการนำมาแสดงบนจอมอนิเตอร์ โดยไม่ต้องทำงานผ่านเครื่องคอมพิวเตอร์ เพื่อการประหยัดพลังงาน

เทคโนโลยีเกี่ยวกับการ์ดหน่วยความจำ (Memory Card) ได้มีการพัฒนาออกสู่ตลาดมากมายหลายรูปแบบ เช่น Memory Stick, Smart media, CF, SD/MMC Card เป็นต้น การใช้งานก็จะมีลักษณะแตกต่างกัน ในส่วนของผู้จัดทำได้เลือกใช้ SD Card ในการจัดเก็บไฟล์ภาพ เนื่องจากเห็นว่ามีความเหมาะสมในเรื่องของราคาที่ไม่แพงมากและหาซื้อได้ตามท้องตลาดทั่วไป

สุดท้ายในส่วนของไมโครคอนโทรลเลอร์ เนื่องจากผู้จัดทำต้องการไมโครคอนโทรลเลอร์ที่มีการประมวลผลทางสัญญาณที่รวดเร็ว และกินกำลังไฟต่ำ จึงได้เลือกใช้ไมโครคอนโทรลเลอร์ตระกูล ARM7 (LPC2119) ในการควบคุมการทำงานทั้งหมด

1.2 วัตถุประสงค์

เพื่อศึกษาการอ่าน-เขียน ข้อมูลจากการ์ดหน่วยความจำแบบ SD Card ศึกษาการเข้ารหัสไฟล์ภาพแบบ Bitmap ศึกษาการติดต่อสื่อสารข้อมูลในลักษณะ SPI Bus ศึกษาเกี่ยวกับการสร้างสัญญาณภาพบนจอมอนิเตอร์ และศึกษาการใช้งานไมโครคอนโทรลเลอร์ตระกูล ARM7

1.3 ขอบเขตของโครงงาน

โครงงานอุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์นี้ ได้ออกแบบและสร้างวงจรในการอ่าน-เขียนการ์ดหน่วยความจำแบบ SD Card ส่วนการถอดรหัสไฟล์ Bitmap จะใช้ไมโครคอนโทรลเลอร์ในการถอดรหัส และมีการสร้างสัญญาณภาพแบบอนาลอกผ่าน Port D-Sub เพื่อนำไปแสดงผลบนจอภายนอก ในการควบคุมการทำงานทั้งหมดจะใช้ไมโครคอนโทรลเลอร์ ARM7 (LPC2119) ในการควบคุมการทำงานทั้งหมด

1.4 โครงสร้างของรายงาน

รายงานฉบับนี้ได้อธิบายขั้นตอน วิธีในการออกแบบ รวมทั้งวงจรและผลการทดลอง ทดสอบคุณสมบัติต่างๆของอุปกรณ์เครื่องนี้ โดยเนื้อหาแบ่งเป็นบทต่างๆดังนี้

บทที่ 2 ทฤษฎี กล่าวถึงทฤษฎีและหลักการพื้นฐานต่างๆ ที่เกี่ยวข้องกับการออกแบบและสร้างอุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

บทที่ 3 การออกแบบ กล่าวถึงขั้นตอนในการออกแบบและการติดต่อกับการ์ดหน่วยความจำแบบ SD Card โดยใช้ไมโครคอนโทรลเลอร์ ARM7

บทที่ 4 การทดลองและผลการทดลอง กล่าวถึงการทดลองและผลการทดลองเมื่อทำการทดสอบสั่งงานอุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์ด้วย Input ต่างๆ

บทที่ 5 สรุปผลการทดลอง และวิจารณ์ผลการทดลอง

บทที่ 2

ทฤษฎี

2.1 ลักษณะเฉพาะของ SD Card

ลักษณะภายนอกของ SD Card นั้นจะมีขนาดที่มีการกำหนดไว้เป็นมาตรฐาน คือ มีความกว้าง 24 mm ความยาว 32 mm และความหนา 2.1 mm มีน้ำหนักโดยประมาณ 2 กรัม

การ์ด SD จะมีขาสัมผัสทั้งหมด 9 ขา ประกอบด้วยขาที่ใช้ต่อกับแหล่งจ่ายไฟ VCC VSS และขาสัญญาณเพื่อใช้ติดต่อกับตัวอุปกรณ์ โดยตัวการ์ด SD จะมีการทำงาน 2 โหมด คือ SD Bus และ SPI Bus ทำให้ขาที่ใช้ในการติดต่อ จะมีหน้าที่การทำงานได้ 2 หน้าที่

2.1.1 วิธีการส่งข้อมูลแบบ SD Bus

การส่งแบบนี้จะใช้สายติดต่อ จำนวน 6 สาย และสายที่ใช้จ่ายไฟอีก 2 สาย ประกอบด้วย

1. CMD: Command is bi-directional signal.(Host and card driver are operating in push pull mode.)
2. DAT0-3: Data lines are bi-directional signals. (Host and card drivers are operating in push pull mode.)
3. CLK: Clock is a host to cards signal. (CLK operates in push pull mode.)
4. *VDD: VDD is the power supply line for all cards*
5. VSS: VSS are two ground lines

ขั้นตอนการตรวจสอบ คำสั่งจะถูกส่งไปที่การ์ดแต่ละตัวโดยจะยอมให้ Application ตรวจสอบการ์ดและระบุ Logical Address เพื่อนำไปสู่ Physical Slot ข้อมูลจะถูกส่งไปที่การ์ดอย่างสม่ำเสมอ อย่างไรก็ดีเพื่อง่ายต่อการควบคุม Card Stack หลังจากเสร็จสิ้นขั้นตอนการตรวจสอบ คำสั่งทั้งหมดจะถูกส่งไปพร้อมๆกันกับทุกการ์ด Address Information จะถูกเตรียมมาใน Command Packet

SD Bus ยอมให้มีการปรับแต่งจำนวนสายส่งข้อมูลหลังจากที่ปรับปรุงเพิ่มประสิทธิภาพ จากปกติ SD Card จะใช้แค่ DAT0 เพื่อการส่งข้อมูลหลังจากขั้นตอนการตรวจสอบ Host สามารถที่จะเปลี่ยนความกว้างของระบบ Bus (จำนวนของสายส่งข้อมูล) ขั้นตอนการติดต่อแบบนี้มีความสมดุลระหว่าง ราคาของอุปกรณ์ และประสิทธิภาพของระบบ

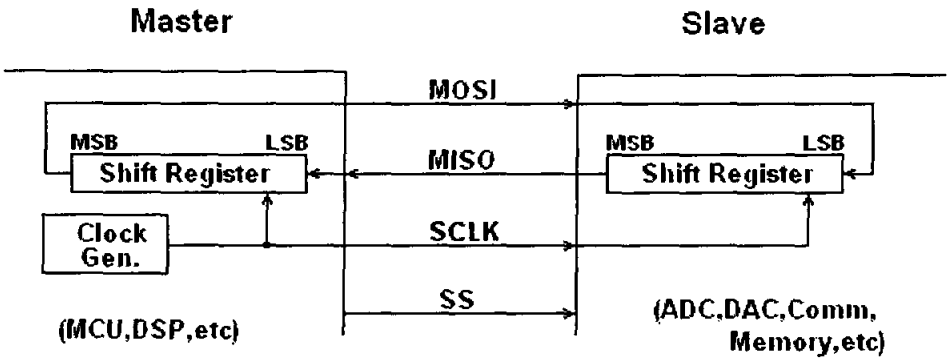
2.1.2 วิธีการส่งข้อมูลแบบ SPI Bus

SD Card แบบ SPI interface มีความเข้ากันได้กับ SPI Hosts ที่มีทั่วไปในตลาด เหมือนอุปกรณ์ที่ติดต่อแบบ SPI อื่นๆ SD card แบบ SPI Channel ประกอบด้วย

1. CS: Host to card Chip Select signal.
2. SCLK: Host to card clock signal.
3. Data In: Host to card data signal.
4. Data Out: Card to host data signal.

โครงสร้างของการเชื่อมต่อแบบ SPI แสดงดังรูปที่ 2.1 ตัว Master และตัว Slave จะติดต่อกันด้วยสายสัญญาณสามเส้น คือ SCLK (Serial Clock), MISO (Master-In Slave-Out) และ MOSI (Master-Out Slave-In) ส่วนสาย SS (Slave Selected) ที่เพิ่มขึ้นมานั้นจะใช้เป็นสายสำหรับเลือกตัว Slave ที่จะมาติดต่อกับตัว Master

ในโหมด SPI นี้ การส่งข้อมูลจะทำการส่ง MSB (Most Significant Bit) ออกไปก่อน



รูปที่ 2.1 โครงสร้างอย่างง่ายของการติดต่อในโหมดSPI

ลักษณะเฉพาะของ SPI Bus ทั้งที่ติดตั้งอยู่ใน SD Card ก็ดี คือการส่งข้อมูลแบบ Byte ข้อมูลทั้งหมดจะถูกแปลงเป็น 8 bit Bytes และจะถูกเรียงเป็นเส้นเดียว คือสัญญาณ CS

DPI Standard จะกำหนด Physical Link เท่ากัน จะไม่เกี่ยวกับการส่ง Data transfer protocol ในโหมด SPI Bus SD card จะถูกใช้เป็นส่วนย่อยของ SD Protocol และเซตคำสั่ง

การยืนยัน SD card และ Addressing Algorithms จะถูกแทนโดยสัญญาณ Hardware Chip Select (CS) การ์ดจะถูกเลือกในทุกๆคำสั่ง โดยการ Active Low ที่ขาสัญญาณ CS

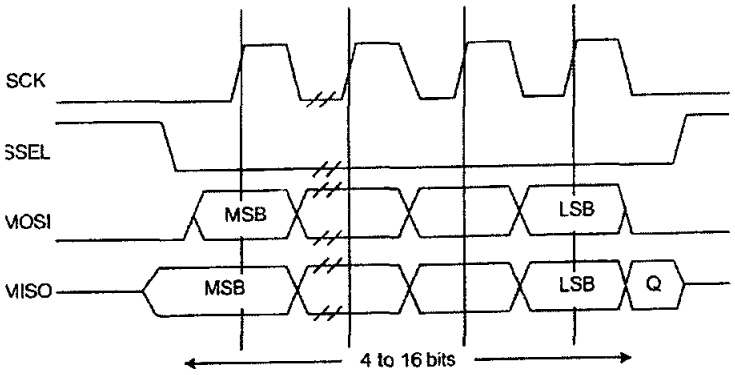
สัญญาณ CS จะถูก Active ต่อเนื่องเป็นช่วงเวลาของ SPI transaction (คำสั่ง, การตอบสนอง และข้อมูล) ข้อยกเว้นข้อเดียวคือ การ์ด Programming Time ที่เวลานี้ Host สามารถที่จะยืนยัน CS Signal โดยที่ไม่ต้องผ่านกระบวนการ Programming

ในระบบ SPI Bus จะไม่มีการกำหนด Address การเลือกอุปกรณ์ตัวที่ต้องการติดต่อ ทำได้ด้วยการส่งสัญญาณไปที่ขา Slave Select ซึ่งเป็นขาแยกต่างหาก ดังนั้นถ้ามีอุปกรณ์ SPI หลายตัวที่ต่อร่วมกันจะต้องต่อขา SCK, MISO, MOSI ขนานกันได้ แต่ขาที่ต่อไปยัง Slave Select จะต้องแยกจากกัน

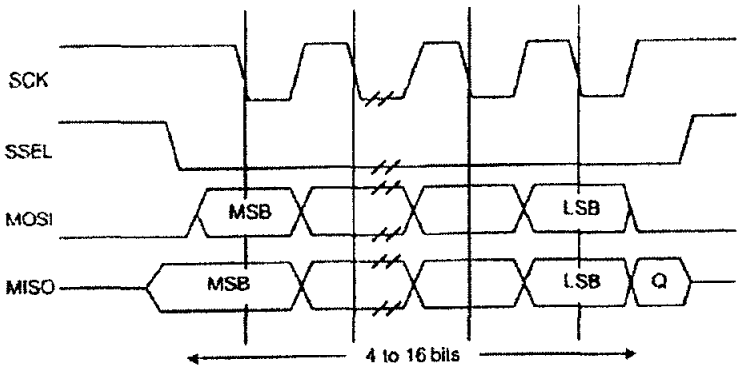
ใน SPI Bus จะแบ่งอุปกรณ์ได้เป็น 2 ประเภท คือ Master และ Slave ถ้าให้ทำงานเป็น Master ทำได้โดยการต่อขา SSEL ให้มีค่า Logic เป็น 1 ถ้าทำงานเป็น Slave ทำได้โดยต่อขา SSEL เข้ากับอุปกรณ์ที่เป็น Master

อุปกรณ์ที่เป็น Master จะเป็นผู้ควบคุม SPI Bus โดยส่งสัญญาณ SCK ไปให้อุปกรณ์ทุกตัว เมื่อต้องการเลือกอุปกรณ์ตัวใด ให้ส่งค่า Logic 0 ไปยังขา SSEL (ในอุปกรณ์ SPI จำนวนมากจะตั้งชื่อขาสัญญาณนี้เป็น CS) ของอุปกรณ์แล้วจึงส่งข้อมูลให้อุปกรณ์ทางขา MOSI ถ้าจะรับข้อมูลจากอุปกรณ์ รับทางขา MISO

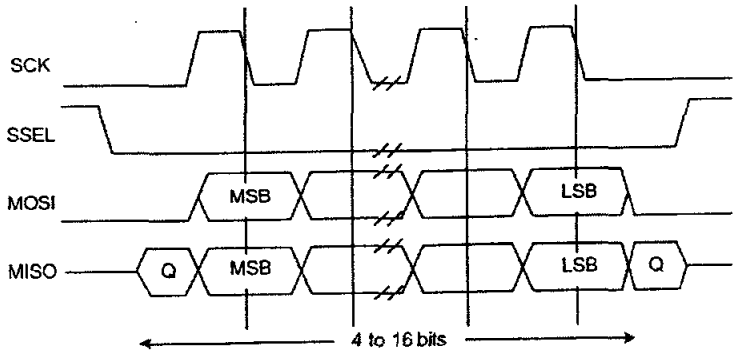
ขั้นตอนต่อมาเป็นการกำหนดการขั้วของสัญญาณนาฬิกา (Clock Polarity: CPOL) ว่าให้ทำงานที่ระดับแรงดัน 1 หรือ 0 และกำหนดเฟสของสัญญาณนาฬิกา ซึ่งกำหนดได้หลายแบบ และจะมีผลต่อการส่งข้อมูล เมื่อกำหนดค่า CPOL และ CPHA ต่างกัน ดังรูป



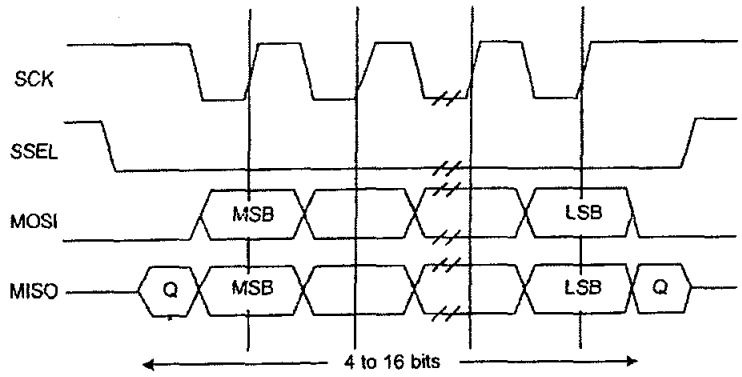
รูปที่ 2.2 รูปแบบเฟรมของ SPI เมื่อ CPOL=0 และ CPHA=0



รูปที่ 2.3 รูปแบบเฟรมของ SPI เมื่อ CPOL=0 และ CPHA=1



รูปที่ 2.4 รูปแบบเฟรมของ SPI เมื่อ CPOL=1 และ CPHA=0



รูปที่ 2.5 รูปแบบเฟรมของ SPI เมื่อ CPOL=1 และ CPHA=1

เมื่อกำหนดค่าตัวแปร CPOL และ CPHA แล้วจะแสดงความสัมพันธ์ ระหว่างสัญญาณ SCK และข้อมูลได้ดังตาราง

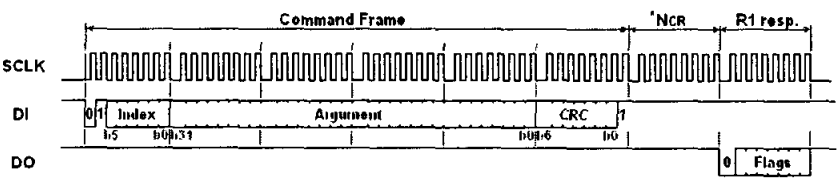
ตารางที่ 2.1 แสดงความสัมพันธ์ ระหว่าง CPHA และข้อมูล

CPOL และ CPHA	ข้อมูลบิตแรกจะมาเมื่อ	ข้อมูลบิตอื่นๆ	สุ่มข้อมูลที่
CPOL = 0, CPHA = 0	ก่อนที่จะพบขอบขาขึ้นของ SCK ตัวแรก	ขอบขาลงของ SCK	ขอบขาขึ้นของ SCK
CPOL = 0, CPHA = 1	ขอบขาขึ้นของ SCK ตัวแรก	ขอบขาขึ้นของ SCK	ขอบขาลงของ SCK
CPOL = 1, CPHA = 0	ก่อนที่จะพบขอบขาลงของ SCK	ขอบขาขึ้นของ SCK	ขอบขาลงของ SCK
CPOL = 1, CPHA = 1	ขอบขาลงของ SCK ตัวแรก	ขอบขาลงของ SCK	ขอบขาขึ้นของ SCK

ในการกำหนดค่าของ CPOL และ CPHA จะต้องกำหนดตามคู่มือของอุปกรณ์ SPI ที่ต้องการติดต่อ

2.1.3 คำสั่งและการตอบกลับ

ในโหมด SPI นั้น โครงสร้างของคำสั่งจะถูกกำหนดไว้ให้มีความยาว 6 Byte ดังแสดงในรูปที่ 2.6 เมื่อชุดคำสั่งถูกส่งไปที่การ์ด จะมีการตอบกลับจากการ์ดไปที่ Host ในรูปแบบ R1, R2 หรือ R3 เนื่องจากการถ่ายโอนข้อมูลนั้นจะถูกขับโดย Clock แบบอนุกรมที่ Host สร้างขึ้น ดังนั้น Host จะต้องสร้าง Clock ต่อไปเรื่อยๆ จนกว่าจะได้รับการตอบกลับจากการ์ด ช่วงระยะเวลาก่อนที่การ์ดจะตอบกลับหลังจากได้รับคำสั่งจาก Host (NCR) คือ 0 ถึง 8 Byte สำหรับ SDC และ 1 ถึง 8 Byte สำหรับ MMC ในช่วงที่มีการโอนถ่ายข้อมูลนี้ ขา SS จะต้องถูกกำหนดให้มีสถานะลอจิกเป็น 0 เสมอ



รูปที่ 2.6 Command Frame ที่ส่งจาก Host ไปหาการ์ด

2.1.4 ชุดคำสั่งที่ใช้ในโหมด SPI

คำสั่งดังแสดงให้เห็นนี้เป็นเพียงคำสั่งโดยทั่วไปที่ใช้สำหรับการอ่าน เขียนและ Initial Card เท่านั้น

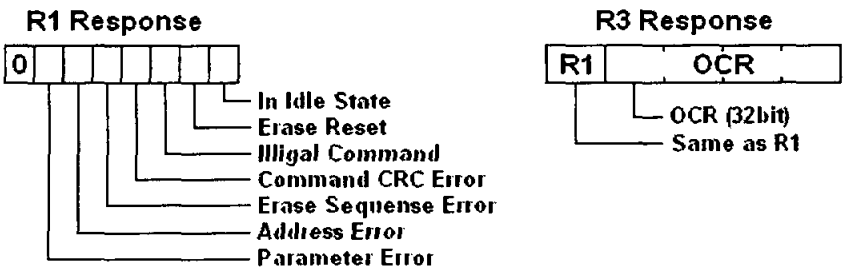
ตารางที่ 2.2 แสดงชุดคำสั่งของโหมด SPI

Command Index	Argument	Response	Data	Abbreviation	Description
CMD0	None(0)	R1	No	GO_IDLE_STATE	Software reset.
CMD1	None(0)	R1	No	SEND_OP_COND	Initiate initialization process.
CMD9	None(0)	R1	Yes	SEND_CSD	Read CSD register.
CMD10	None(0)	R1	Yes	SEND_CID	Read CID register.
CMD12	None(0)	R1b	No	STOP_TRANSMISSION	Stop to read data.
CMD17	Address[31:0]	R1	Yes	READ_SINGLE_BLOCK	Read a block.
CMD18	Address[31:0]	R1	Yes	READ_MULTIPLE_BLOCK	Read multiple blocks.
CMD23	Number of blocks[15:0]	R1	No	SET_BLOCK_COUNT	For only MMC. Define number of blocks to transfer with next multi-block read/write command.
ACMD23 (*1)	Number of blocks[22:0]	R1	No	SET_WR_BLOCK_ERASE_ COUNT	For only SDC. Define

					number of blocks to pre-erase with next multi-block write command.
CMD24	Address[31:0]	R1	Yes	WRITE_BLOCK	Write a block.
CMD25	Address[31:0]	R1	Yes	WRITE_MULTIPLE_BLOCK	Write multiple blocks.
CMD55	None(0)	R1	No	APP_CMD	Application specific command.
CMD58	None(0)	R3	No	READ_OCR	Read OCR.
*1: ACMD <n> means a command sequence of CMD55-CMD <n>.					

2.1.5 Command Response ในโหมดSPI

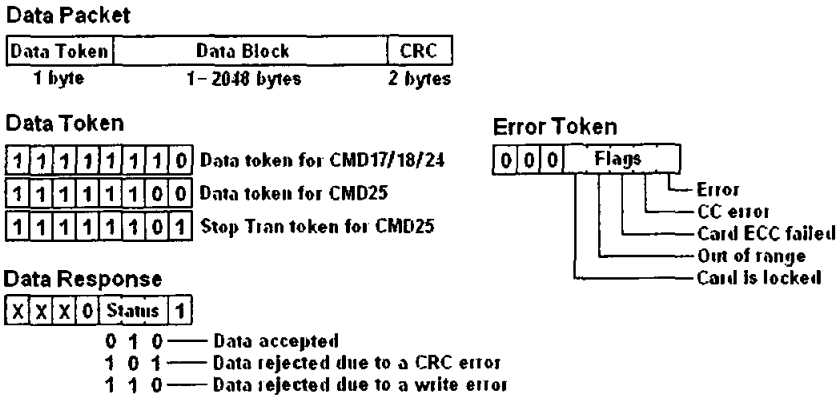
การตอบกลับในโหมด SPI จะมีอยู่สามรูปแบบ คือ R1, R2 และ R3 ดังรูปที่ 2.7 ขึ้นอยู่กับคำสั่งที่ Host ส่งมา แต่คำสั่งส่วนมากจะมีการตอบกลับในรูปแบบ R1 ถ้าหาก R1 มีค่าเป็น 0x00 แสดงว่าไม่มีข้อผิดพลาดเกิดขึ้น ส่วนการตอบสนองแบบ R3 นั้นจะตอบกลับเฉพาะ CMD58 เท่านั้น



รูปที่ 2.7 การตอบกลับแบบต่างๆ

2.2 การถ่ายโอนข้อมูล

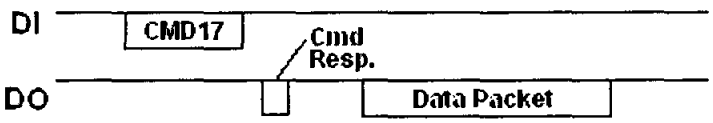
2.2.1 Data Packet และ Data response



รูปที่ 2.8 โครงสร้างของData Packet

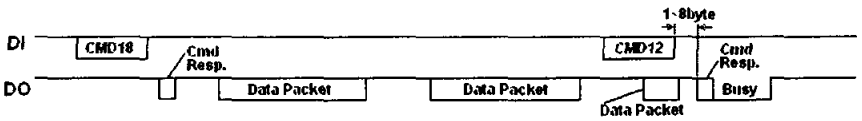
ในการถ่ายโอนข้อมูล จะมีการถ่ายโอนหลังจากมีการส่ง Command Response แล้ว บล็อกข้อมูลจะถูกถ่ายโอนในรูป Data Packet ซึ่งประกอบไปด้วย Data Token, Data Block และ CRC ดังแสดงในรูปที่ 2.8 จะเห็นได้ว่า Data Token จะมีอยู่สามแบบ ซึ่งขึ้นอยู่กับแต่ละคำสั่งที่ส่ง

2.2.2 คำสั่ง Single Block Read



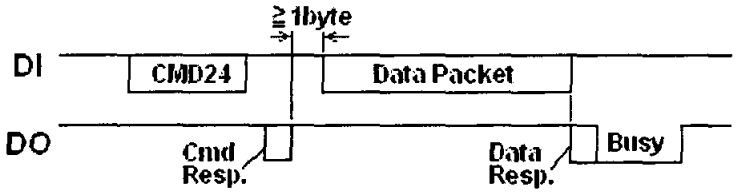
Argument ที่ต่อท้ายคำสั่งนี้จะเป็นตัวกำหนดแอดเดรสเริ่มต้นของข้อมูลที่จะอ่าน เมื่อคำสั่งนี้ได้รับการตอบกลับ กระบวนการอ่านข้อมูลก็จะเริ่มขึ้นโดยชุดข้อมูลที่ต้องการอ่านนั้นก็จะถูกส่งไปยัง Host เมื่อ Host ตรวจพบ Data Token ที่ส่งมา Host ก็จะทำการอ่านบล็อกข้อมูลที่ตามหลัง Token นั้น เมื่อมีความผิดพลาดเกิดขึ้นในการทำงาน Token ที่แสดงความผิดพลาดจะถูกส่งมาแทนที่บล็อกข้อมูล

2.2.3 คำสั่ง Multiple Block Read



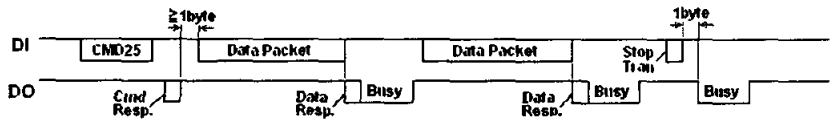
คำสั่งนี้เป็นการอ่านข้อมูลครั้งละหลายๆ บล็อกจากตำแหน่งแอดเดรสที่กำหนด ถ้าหากไม่มีการกำหนดจำนวนบล็อกในการอ่านต่อหนึ่งครั้งไว้ กระบวนการอ่านนี้จะดำเนินต่อเนื่องไปเรื่อยๆ จนกว่าจะมีการส่งคำสั่ง CMD12 ไปที่การ์ด การอ่านข้อมูลนี้จึงจะหยุดลง

2.2.4 คำสั่ง Single Block Write



เมื่อคำสั่งนี้ได้รับการตอบรับ Host ก็จะทำการส่งแพ็คเกจข้อมูลไปที่การ์ดหลังจากนั้นเป็นช่วงประมาณ 1 Byte แพ็คเกจของข้อมูลนี้ก็จะมีลักษณะเหมือนกับแพ็คเกจของข้อมูลในคำสั่ง Read เมื่อข้อมูลถูกส่งไปที่การ์ดแล้ว การ์ดก็จะตอบกลับมาด้วย Data Response ในทันที การ์ดโดยส่วนใหญ่จะสามารถเขียนข้อมูลได้บล็อกละ 512 Byte

2.2.5 คำสั่ง Multiple Block Write



คำสั่งนี้จะเขียนข้อมูลครั้งละหลายๆ บล็อกลงไปในแอดเดรสที่เรากำหนด ถ้าหากไม่มีการกำหนดจำนวนบล็อกในการถ่ายโอน การถ่ายโอนนี้ก็จะดำเนินต่อไปเรื่อยๆ จนกว่าจะถูกหยุดโดย Stop Tran Token ซึ่งก็จะทำให้เกิด Busy Flag ขึ้นเป็นช่วงระยะ 1 Byte ต่อจาก Stop Tran Token

2.2.6 ขั้นตอนการทำงานเมื่อเป็น Master

เมื่อกำหนดให้ Microcontroller เป็น Master จะมีขั้นตอนต่างๆ ในการทำงานดังนี้

1. กำหนดค่า Register SPI Clock Counter ให้สัญญาณนาฬิกามีค่าตามต้องการ
2. กำหนดค่า Register ควบคุมการทำงานของ SPI ให้มีค่าตามที่ต้องการ
3. เขียนข้อมูลที่ต้องการส่งไปยัง Register เก็บข้อมูลของ SPI การเขียนข้อมูลนี้จะสั่งให้เริ่มต้นส่งข้อมูล
4. เมื่อส่งข้อมูลครบทุกบิตแล้ว บิต SPIF ของ SPI Status Register จะมีค่าเป็น 1
5. อ่านค่าสถานะของ SPI จาก SPI Status Register
6. อ่านข้อมูลที่ได้รับจาก SPI Data Register
7. กลับไปยังข้อ 3 ถ้ามีข้อมูลที่ต้องการส่งอีก

หมายเหตุ : ในการล้างค่าบิต SPIF ทำได้โดยการอ่านหรือเขียนค่าไปยัง SPI Data Register

2.2.7 การกำหนดค่าเริ่มต้นสำหรับการติดต่อกับอุปกรณ์ผ่าน SPI0

ภายในไมโครคอนโทรลเลอร์ตระกูล LPC2000 มีวงจรสำหรับติดต่อกับบัส SPI จำนวนสองวงจร คือ SPI0 และ SPI1 โดย SPI1 มีชื่อเรียกเฉพาะว่า Synchronous Serial Port (SSP)

หลังจากที่สั่งที่ Register PINSEL0 ให้ขาที่เกี่ยวข้องมีการทำงานเป็น SPI0 แล้ว ขั้นต่อมาจะต้องกำหนดค่าให้กับ Register ที่เกี่ยวข้องกับ SPI0 ซึ่งมีทั้งหมด 5 ตัวดังตาราง

ตารางที่ 2.3 Register ที่เกี่ยวข้องกับ SPI0 ทั้ง 5 ตัว

ชื่อ	ความหมาย	การติดต่อ	ค่าหลังรีเซท	Address
S0SPCR	SPI Control Register ใช้ควบคุมการทำงานของ SPI	อ่าน/เขียน	0x00	0xE002 0000
S0SPSR	SPI Status Register แสดงสถานการณ์ทำงานของ SPI	อ่าน	0x00	0xE002 0004
S0SPDR	SPI Data Register เมื่อต้องการส่งข้อมูล ให้เขียนค่ามายัง Register นี้ในโหมดการรับข้อมูล ให้อ่านค่าจาก Register นี้	อ่าน/เขียน	0x00	0xE002 0008
S0SPCCR	SPI Clock Counter Register ใช้ควบคุมค่าความถี่ SCK0 ของ Master	อ่าน/เขียน	0x00	0xE002 000C
S0SPINT	SPI Interrupt Flag เก็บ Interrupt Flag ของ SPI	อ่าน/เขียน	0x00	0xE002 001C

2.3 ประเภทของไฟล์ภาพ

2.3.1 BMP (Bitmap)

Bitmap คือคำที่ใช้เรียกข้อมูลภาพที่เก็บอยู่ในหน่วยความจำซึ่งเป็นภาพที่เก็บในลักษณะ Pixel ต่อ Pixel โดยเก็บข้อมูลสีของแต่ละ Pixel เรียงกันไปในหน่วยความจำ ข้อมูลสีของแต่ละ Pixel จะมีขนาดที่ใช้ในการเก็บขึ้นอยู่กับความละเอียดสีที่ใช้ เช่นถ้าเป็น Bitmap แบบ 256 สี 1 Pixel จะมีขนาด 8 Bit (2 กำลัง 8) ดังนั้น 1 Pixel ของ Bitmap 256 สี จะมีขนาด 1 Byte โดยลักษณะของ Bitmap 256 สี ก็เป็น Array ของข้อมูลขนาด 1 Byte ในลักษณะเดียวกัน ถ้าเป็น 32 Bit ต่อ Pixel จะได้ว่า 1 Pixel มีขนาด 4 Byte ต่อ Pixel และจะเก็บได้ 2 ยกกำลัง 32 สี

เนื้อที่ที่จะต้องใช้ในการเก็บ Bitmap ในหน่วยความจำจะเป็นผลคูณของ Byte ต่อ Pixel, Width และ Height ของ Bitmap ตัวอย่างเช่น Bitmap แบบ 256 สี กว้าง 640 สูง 480 จะมีขนาดประมาณ 640 x 480 x 1 Byte

เนื่องจาก Bitmap ที่เก็บอยู่ในหน่วยความจำนั้นจะเป็นข้อมูล Pixel เรียงต่อกันไปเรื่อยๆ จนหมดภาพ ดังนั้นจึงต้องมีการกำหนดความกว้างและความยาวของ Bitmap ด้วย เพื่อให้โปรแกรมส่วนที่นำ Bitmap ไปใช้ทราบว่า Bitmap มีความกว้าง – ยาว เท่าไหร่

2.3.2 JPEG (Joint Photographic Expert Group)

JPEG คือรูปแบบการบีบอัดแฟ้มภาพแบบสูญเสีย โดยยังให้เสียความละเอียดน้อยที่สุด รูปแบบแฟ้มสำหรับวิธีการนี้ได้แก่ .jpeg, .jpg, .jpe, .jfif, .jfi (อาจจะเป็นตัวเล็กหรือตัวใหญ่ก็ได้) รูปแบบแฟ้ม JPEG

นี้ เป็นรูปแบบแฟ้มที่ใช้กันในการจัดเก็บและแลกเปลี่ยนรูปภาพบนอินเตอร์เน็ตมากที่สุด โดยเฉพาะภาพถ่าย เนื่องจากสามารถเก็บความละเอียดสูงได้โดยใช้ขนาดไฟล์ที่เล็ก สามารถเก็บภาพสีได้หลากหลายระดับความแม่นยำของสี (Bit Depth) ความสามารถในการย่อขนาดไฟล์ของแฟ้ม JPEG นั้นเกิดจากการใช้เทคนิคการย่อขนาดภาพแบบการบีบอัดคงข้อมูลหลัก (Lossy Compression) หรือการบีบอัดแบบมีความสูญเสียทำให้ไม่นิยมใช้กับภาพที่เป็น ลายเส้น หรือไอคอนต่างๆ เนื่องจากจะไม่ได้ประสิทธิภาพเท่าการเก็บในรูปแบบอื่น อย่าง PNG หรือ GIF การบีบอัดของ JPEG นั้นจะใช้เทคนิคที่เรียกว่า DCT (Discrete Cosine Transform) ซึ่งเป็นการแปลงค่าความสว่างของภาพให้อยู่ในรูปแบบเชิงความถี่ (Frequency Domain) ทำให้สามารถเลือกแทนค่าของสัมประสิทธิ์หรือในที่นี้คือแอมพลิจูดของค่าความถี่ต่างๆ ได้โดยอาศัยตัวแปรที่มีนัยสำคัญที่ต่างกันได้ การที่สามารถลดนัยสำคัญของค่าตัวเลขลงไปได้ทำให้สามารถลดขนาดของหน่วยความจำหรือขนาดไฟล์ที่ใช้เก็บตามไปได้

2.3.3 GIF (Graphics Interchange Format)

มีนามสกุล (file extension) คือ .gif เป็นวิธีการเก็บไฟล์ภาพแบบบีบอัดคล้ายกับ JPEG โดยทั่วไปแล้วไม่สามารถเก็บภาพที่ถ่ายจากธรรมชาติได้มีขนาดเล็กเท่ากับแบบ JPEG แต่สามารถเก็บภาพ เช่น ภาพการ์ตูน ได้เป็นอย่างดี นอกจากนี้ GIF ยังสามารถเก็บภาพไว้ได้หลายๆภาพ ในไฟล์เดียว จึงถูกนำไปใช้สร้างภาพเคลื่อนไหวง่ายๆ เช่น ในอินเตอร์เน็ต ในการบีบอัดของ GIF จะเป็นประเภทไม่สูญเสีย (lossless compression) โดยรองรับภาพที่มีสีสูงสุด 256 สี ด้วยเหตุนี้มาตรฐาน GIF จึงไม่ความต่อเนื่องของสี คือ นิยมใช้กับภาพถ่ายซึ่งเป็นรูปที่มีจำนวนสีมากกว่าและมีความต่อเนื่องของสี

2.4 ตารางการจัดเรียงไฟล์ (File Allocation Table: FAT)

ระบบปฏิบัติการของคอมพิวเตอร์แต่ละ Platform จะมีการจัดระบบไฟล์ในฮาร์ดดิสก์ที่แตกต่างกัน บางระบบสามารถใช้ระบบไฟล์ได้หลายรูปแบบ โดยระบบไฟล์นั้นเป็นตารางที่ใช้บอกตำแหน่งของข้อมูลต่างๆ ที่อยู่บนฮาร์ดดิสก์ว่าอะไรอยู่ตรงไหน ปกติเมื่อซื้อฮาร์ดดิสก์มาใหม่ต้องทำการจัดข้อมูล (Format) ให้ฮาร์ดดิสก์ก่อนที่จะนำไปบรรจุข้อมูล การจัดข้อมูลในฮาร์ดดิสก์เป็นการแบ่งฮาร์ดดิสก์ออกเป็นส่วนๆ เพื่อให้คอมพิวเตอร์รู้ว่าตำแหน่งของข้อมูลอยู่ตรงไหน

FAT เป็นระบบไฟล์ที่ใช้ในระบบปฏิบัติการของ Microsoft และเป็นระบบไฟล์ที่มีการพัฒนาอย่างต่อเนื่อง โดยจะมีลักษณะคือ เป็นการกำหนดหมายเลขให้กับทุกๆ Cluster ในแต่ละส่วนแบ่งของฮาร์ดดิสก์แล้วทำการสร้างตารางที่มีจำนวนช่อง ตามจำนวนของ Cluster เพื่อเป็นการระบุตำแหน่งหรือ Cluster ที่ทำการเก็บข้อมูลของไฟล์แต่ละไฟล์ และมีตารางอีกตารางที่เรียกว่า Directory สำหรับเก็บข้อมูล รายละเอียดของไฟล์ เช่น Attribute ต่างๆ และหมายเลข Cluster เริ่มต้นที่เก็บตัวข้อมูลจริงๆ

ระบบจัดการไฟล์แบบ FAT เป็นระบบที่ไม่ยุ่งยากซับซ้อน ดังนั้นจึงถูกรองรับ จากระบบปฏิบัติการที่มีอยู่สำหรับคอมพิวเตอร์ส่วนบุคคล และยังสะดวกในการแลกเปลี่ยนข้อมูลระหว่างระบบปฏิบัติการที่แตกต่างกันซึ่งถูกติดตั้งบนคอมพิวเตอร์เครื่องเดียวกัน

ข้อด้อยของระบบจัดการไฟล์แบบ FAT คือ เมื่อไฟล์ถูกลบ และไฟล์ใหม่ถูกเขียนลงไป Fragment ของแต่ละไฟล์จะมีโอกาสกระจัดกระจายออกไปอยู่ทั่วทั้งหน่วยความจำ ส่งผลให้การอ่านและการเขียนไฟล์ทำได้ช้า การจัดเรียงข้อมูลเป็นหนึ่งในวิธีการทำให้การจัดเรียงไฟล์แบบ FAT เป็นระเบียบแต่จะต้องทำการจัดเรียงข้อมูลอยู่บ่อย และเป็นกระบวนการที่ใช้เวลามาก

ระบบไฟล์ FAT มีหลายรุ่นดังต่อไปนี้

2.4.1 FAT12

ระบบ FAT12 เป็นรุ่นที่เก่าแก่ที่สุดของตระกูล FAT ใช้กับระบบปฏิบัติการ Dos ซึ่งใช้ 12 bits ในการอ้างอิงถึงหมายเลข Cluster เพราะฉะนั้นสามารถอ้างถึง Cluster ได้มากสุด 4086 Cluster ระบบ FAT12 จึงเหมาะกับหน่วยความจำที่มีเนื้อที่ไม่มาก หรือมีเนื้อที่ไม่เกิน 16 MB และระบบไฟล์ชนิดนี้สามารถใช้งานกับไฟล์ที่มีชื่อยาวเพียง 8.3 ตัวอักษรเท่านั้น

2.4.2 FAT16

สามารถใช้งานร่วมกับระบบปฏิบัติการที่หลากหลายได้ เช่น Dos Windows95, 98, ME ซึ่งใช้ 16 bits เพื่ออ้างถึงหมายเลข Cluster เพราะฉะนั้นสามารถอ้างถึงได้ทั้งหมด 65526 Cluster FAT16 นั้นถูกออกแบบมาเพื่อใช้งานกับไฟล์ต่างๆ บนดิสก์ขนาดเล็ก และขนาดกลาง ซึ่งมีเนื้อที่ประมาณ 2048 MB

ปัญหาของ FAT16 คือ

1. ระบบ FAT16 ใช้งานพื้นที่ของหน่วยความจำสิ้นเปลืองมาก เพราะจำนวนสูงสุดของ Cluster ต่อ Partition นั้นถูกกำหนดเอาไว้ตายตัว (65526 Cluster) ดังนั้นเมื่อหน่วยความจำมีขนาดที่ใหญ่ขึ้น แต่ว่าจำนวน Cluster ยังเท่าเดิม ก็หมายความว่าขนาดของ Cluster จะใหญ่ขึ้นไปด้วย อย่างในกรณีของหน่วยความจำ ขนาด 2 GB นั้นจะมี Cluster ที่ใหญ่ถึง 32 KB นั้นก็หมายความว่า ต่อให้พิมพ์เอกสารที่มีตัวอักษรเพียง 10 ตัว แต่ขนาดของไฟล์ก็จะมีถึง 32 KB ซึ่งเป็นขนาดเล็กสุดของ Cluster เลยทีเดียว
2. ขีดจำกัดเรื่องขนาดสูงสุดของหน่วยความจำที่รองรับได้เพราะเริ่มต้น FAT16 ถูกออกแบบมาเพื่อใช้งานกับดิสก์ที่มีขนาดเล็ก ดังนั้นในยุคแรกๆ จึงมีปัญหาตามมา เมื่อระบบ FAT16 ใน MS-Dos ยุคแรก สามารถรองรับหน่วยความจำได้เพียง 32 MB เท่านั้น แต่ต่อมาถูกแก้ไขให้รองรับได้เป็น 128 MB และเรื่อยมาเป็น 2 GB ในปัจจุบัน
3. ระบบ FAT16 สามารถใช้งานกับไฟล์ที่มีชื่อยาวเพียง 8.3 ตัวอักษรเท่านั้น เช่นเดียวกับ FAT12 แต่ได้รับการปรับปรุงให้มีความสามารถมากขึ้นในเวลาต่อมาเพื่อให้สามารถใช้งานกับไฟล์ที่มีชื่อยาวได้ไม่เกิน 256 ตัวอักษร เรียก FAT16 รุ่นนี้ว่า Virtual FAT หรือ VFAT

2.4.3 FAT32

ระบบ FAT32 ใช้กับระบบปฏิบัติการ Windows 95, 98, ME, 2000 โดยทำการแก้ไขเพิ่มเติมจาก FAT16 เพื่อที่จะได้มีจำนวน Cluster ต่อ Partition มากขึ้น ดังนั้นจึงมีความสามารถที่จะรองรับหน่วยความจำ

ที่มีขนาดใหญ่สูงสุด ได้ถึง 2 TB ซึ่งจะใช้ 28 bits (อีก 4 bits สำรองเอาไว้) ฉะนั้นสามารถอ้างถึง Cluster ได้ทั้งหมด 286 ล้าน Cluster และระบบ FAT32 ยังสามารถรองรับชื่อไฟล์แบบยาว คือ 255 ตัวอักษรได้อีกด้วย

ตารางที่ 2.4 เปรียบเทียบระหว่าง FAT12, FAT16 และ FAT32

	FAT12	**FAT16**	**FAT32**
Developer	Microsoft		
Full Name	File Allocation Table		
	(12-bit version)	(16-bit version)	(32-bit version)
Introduced	1977 (Microsoft Disk BASIC)	July 1988 (MS-DOS 4.0)	August 1996 (Windows 95 OSR2)
Structures			
Directory contents	Table		
File allocation	Linked List		
Bad blocks	Linked List		
Limits			
Max file size	32 MB	2 GB	4 GB
Max number of files	4,077	65,517	268,435,437
Max file name size	8.3 or 255 when using LFNs		
Max volume size	32 MB	2 GB 4 GB	2 TB
Features			
Dates recorded	Creation, modified, access		
Date range	January 1, 1980 – December 31, 2107		
Forks	Not natively		
Attributes	Read-only, hidden, system, volume label, subdirectory, archive		
Permissions	No		
Transparent compression	Per-volume, Stacker, DoubleSpace, Drive Space		No
Transparent encryption	Per-volume only with DR-DOS		No

2.4.4 โครงสร้างของดิสก์หลัก (Main Disk Structure)

Boot sector	More reserved sectors (optional)	File Allocation Table#1	File Allocation Table#2	Root Directory	Data Region (for files and directories)...(To end of partition or disk)

ระบบไฟล์แบบ FAT ประกอบไปด้วย 4 ส่วนที่แตกต่างกัน ดังนี้

1. เซคเตอร์สำรอง (Reserved sectors) อยู่ในส่วนต้นๆ โดย Reserved sector แรกสุดเป็น Boot Sector ซึ่งรวมพื้นที่ที่เรียกว่า BIOS Parameter Block โดยปกติจะประกอบด้วยโค้ดเริ่มต้นของระบบปฏิบัติการ ซึ่งจำนวนของ Reserved sector ทั้งหมดจะถูกระบุอยู่ในส่วนนี้ ข้อมูลสำคัญต่างๆ จาก Boot Sector สามารถเข้าถึงผ่านโครงสร้างระบบปฏิบัติการ ที่เรียกว่า Drive Parameter Block ใน DOS และ OS/2

2. บริเวณตารางจัดเรียงไฟล์ (FAT Region) ส่วนนี้ประกอบไปด้วยตารางการจัดเรียงไฟล์ 2 ชุด ชุดหนึ่งสำหรับใช้งานจริงและอีกชุดเป็นการสำรองไว้ใช้ในเวลาจำเป็น โดยบริเวณตารางจัดเรียงไฟล์นี้จะสอดคล้องกับ บริเวณข้อมูล (Data Region) ทำหน้าที่ระบุตำแหน่ง Cluster ของไฟล์และ Directory ต่างๆ

3. บริเวณไดเรคทอรี่ (Root Directory Region) นี่คือตาราง Directory ซึ่งเก็บข้อมูลเกี่ยวกับ File และ Directory ต่างๆ ในระบบจัดการไฟล์แบบ FAT12 และ FAT16 ตาราง Directory นี้จะอยู่ในบริเวณ Directory เท่านั้น และมีขนาดสูงสุดที่แน่นอน แต่ในระบบจัดการไฟล์แบบ FAT32 ตาราง Directory จะอยู่รวมกับไฟล์ต่างๆ ในบริเวณข้อมูล ซึ่งสามารถขยายขนาดออกไปได้ไม่จำกัด

4. บริเวณข้อมูล เป็นส่วนที่ไฟล์ข้อมูลอยู่จริง ขนาดของ File และ Directory ย่อยสามารถเพิ่มขึ้นได้เท่าที่มี Cluster เหลืออยู่

2.4.4.1 Boot Sector และโครงสร้าง BPB (Bios Parameter Block)

ตารางที่ 2.5 โครงสร้างพื้นฐาน 36 Byte แรกของ Boot Sector ซึ่งใช้ในทุกเวอร์ชันของ FAT

Byte Offset	ความยาว (Byte)	รายละเอียด
0x00	3	คำสั่งกระโดด (เพื่อจะข้ามส่วนหัวของการบูท)
0x03	8	ชื่อ OEM ค่าพื้นฐานคือ ไอบีเอ็ม 3.3 และ MS-DOS 5.0
0x0b	2	จำนวน Byte ต่อ Sector, บล็อกของพารามิเตอร์ BIOS เริ่มต้นที่นี่
0x0d	1	จำนวน Sector ต่อ Cluster
0x0e	2	Reserved Sector
0x10	1	จำนวนของตารางการจัดเรียงข้อมูล

0x11	2	จำนวนสูงสุดของ Directory
0x13	2	จำนวน Sector ทั้งหมด
0x15	1	ตัวบรรยาย Media 0xF8 ด้านเดียว, 80 แทรกต่อด้าน, 9 Sector ต่อแทรก 0xF9 สองด้าน, 80 แทรกต่อด้าน, 9 Sector ต่อแทรก 0xFA ด้านเดียว, 80 แทรกต่อด้าน, 8 Sector ต่อแทรก 0xFB สองด้าน, 80 แทรกต่อด้าน, 8 Sector ต่อแทรก 0xFC ด้านเดียว, 40 แทรกต่อด้าน, 9 Sector ต่อแทรก 0xFD สองด้าน, 40 แทรกต่อด้าน, 9 Sector ต่อแทรก 0xFE ด้านเดียว, 40 แทรกต่อด้าน, 8 Sector ต่อแทรก 0xFF สองด้าน, 40 แทรกต่อด้าน, 8 Sector ต่อแทรก
0x16	2	จำนวน Sector ต่อตารางการจัดเรียงข้อมูล
0x18	2	จำนวน Sector ต่อ Track
0x1a	2	จำนวนของ Head
0x1c	4	จำนวนของ Sector ที่โดนซ่อน
0x20	4	จำนวนของ Sector ทั้งหมด (ใช้ในกรณีมากกว่า 65535 Sector)

ตารางที่ 2.6 โครงสร้างของ Boot Sector ที่เพิ่มขึ้นมาใน FAT32

Byte Offset	**ความยาว (Byte)**	**รายละเอียด**
0x24	4	จำนวน Sector ต่อตารางการจัดเรียงข้อมูล
0x28	2	เครื่องหมายของตารางการจัดเรียงข้อมูล
0x2a	2	เวอร์ชัน
0x2c	4	หมายเลข Cluster ของ Directory เริ่มต้น
0x30	2	หมายเลข Sector ของ Sector ข้อมูล FS
0x32	2	หมายเลข Sector ของสำเนา Boot Sector
0x34	12	สำรอง
0x40	1	Physical Drive Number
0x41	1	สำรอง
0x42	1	Signature
0x43	4	หมายเลข Serial
0x47	11	Volume Label
0x52	8	ชนิดของระบบไฟล์แบบ FAT32

0x5a	420	Code Boot ระบบปฏิบัติการ
0x1FE	2	Last Sector (0x55 0xAA)

2.4.4.2 ตาราง Directory

ตาราง Directory เป็นไฟล์ชนิดพิเศษที่แสดง Directory (ปัจจุบันรู้จักกันในนามของ Folder) แต่ละไฟล์หรือ Directory ประกอบไปด้วย 32 Byte แต่ละ Byte จะบันทึกชื่อไฟล์, นามสกุลไฟล์, คุณลักษณะของไฟล์ (ชนิดเอกสารสำคัญ, Directory, ถูกซ่อน, อ่านได้อย่างเดียว, ระบบ และความจุ), วัน เวลาที่ไฟล์ถูกสร้าง Address ของ Cluster แรกของไฟล์หรือ Directory นั้น และสุดท้ายคือขนาดของไฟล์หรือ Directory

จำนวน Directory ทั้งหมดในบริเวณ Directory และใน Directory ย่อยมีรูปแบบดังนี้

ตารางที่ 2.7 โครงสร้างของตาราง Directory

Byte Offset	**ความยาว (Byte)**	**รายละเอียด**		
0x00	8	0x00		
0x08	3	เครื่องหมายของตารางจัดเรียงข้อมูล		
0x0b	1	**Bit**	**Mask**	**รายละเอียด**
		0	0x01	อ่านได้อย่างเดียว
		1	0x02	ถูกซ่อน
		2	0x04	ระบบ
		3	0x08	Volume Label
		4	0x10	Directory ย่อย
		5	0x20	เอกสารสำคัญ
		6	0x40	อุปกรณ์
		7	0x80	ไม่ใช้
0x0c	1	สำรอง ถูกใช้โดย NT		
0x0d	1	เวลาสร้างไฟล์, ความละเอียด 10 มิลลิวินาที ค่าอยู่ระหว่าง 0-199		
0x0e	2	**Bit**	**รายละเอียด**	
		15-11	ชั่วโมง (0-23)	
			นาที (0-59)	
		10-5	วินาที/2 (0-29)	
		4-0		
0x10	2	**Bit**	**รายละเอียด**	

		15-9 8-5 4-0	ปี (0=1980, 127=2107) เดือน (1= มกราคม, 12=ธันวาคม) วันที่ (1-31)
0x12	2	วันที่ใช้งานไฟล์ล่าสุด ดู Byte Offset 0x10 ประกอบ	
0x14	2	ดัชนี EA (ถูกใช้โดย OS/2 และ NT) ใน FAT12 และ FAT16 หรือ 2 Byte สูงสุดของ Cluster แรกใน FAT32	
0x16	2	เวลาที่แก้ไขไฟล์ล่าสุด ดู Byte Offset 0x0e ประกอบ	
0x18	2	วันที่แก้ไขไฟล์ล่าสุด ดู Byte Offset 0x10 ประกอบ	
0x1a	2	Cluster แรกใน FAT12 และ FAT 16 หรือ 2 Byte ต่ำของ Cluster แรกใน FAT32	
0x1c	4	ขนาดไฟล์	

2.4.4.3 ตาราง FAT

ขนาดของแต่ละ Cluster ขึ้นอยู่กับชนิดของระบบไฟล์แบบ FAT ที่ใช้และขนาดของ Partition โดยปกติขนาดของ Cluster จะอยู่ระหว่าง 2 KB ถึง 32 KB แต่ละไฟล์อาจจะกินเนื้อที่มากกว่า 1 Cluster ขึ้นอยู่กับขนาดของไฟล์ แต่ไม่จำเป็นว่าจะต้องเป็น Cluster ที่ติดกัน

ตารางการจัดเรียงข้อมูลเป็นสมุดรายชื่อของไฟล์ที่สัมพันธ์กับตำแหน่ง Cluster ของไฟล์ แต่ละ Cluster ใน FAT จะบันทึกข้อมูล 1 ใน 5 ดังนี้

- Address ของ Cluster ถัดไปของไฟล์
- สถานะแสดงจุดสิ้นสุดของไฟล์
- สถานะแสดงว่าเป็น Bad Sector
- สถานะแสดงว่าเป็น Cluster ที่ถูกสำรองไว้
- ศูนย์เพื่อแสดงว่าเป็น Cluster ที่ไม่ถูกใช้

แต่ละเวอร์ชันของระบบไฟล์แบบตารางการจัดเรียงข้อมูล จะมีขนาดของ Cluster ในตารางการจัดเรียงข้อมูลที่ต่างกัน ซึ่งขนาดจะถูกระบุโดยชื่อของแต่ละเวอร์ชัน เช่น ระบบไฟล์ FAT16 Cluster จะมีขนาด 16 Bit ในขณะที่ FAT32 จะมีขนาด 32 Bit ตามขนาดของ Partition ที่ใหญ่ขึ้น ซึ่ง FAT32 จะมีประสิทธิภาพมากกว่าในกรณี FAT16 เนื่องจาก FAT32 สามารถแบ่ง Cluster ให้มีขนาดที่เล็กกว่าซึ่งหมายความว่าจะมีพื้นที่สูญเปล่าน้อยกว่าในกรณีของ FAT16

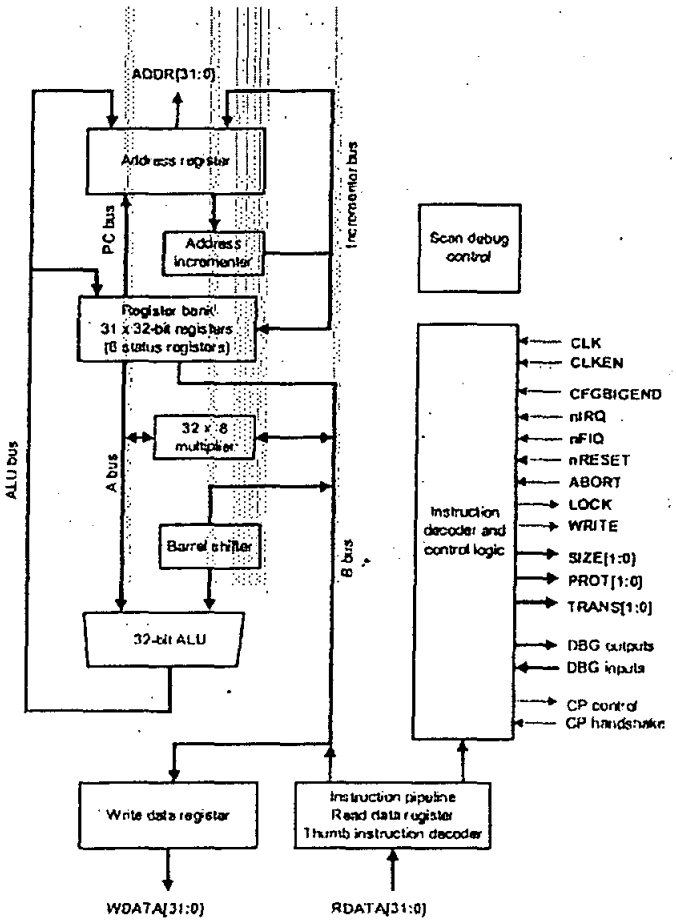
ตารางที่ 2.8 ค่าต่างๆ ในตารางการจัดเรียงข้อมูล

FAT12	FAT16	FAT32	รายละเอียด
0x000	0x0000	0x?0000000	Cluster ว่าง

0x001	0x0001	0x?0000001	Reserved Cluster
0x002 – 0xFEF	0x0002 – 0xFFEF	0x?0000002 – 0x?FFFFFEF	Cluster ที่ถูกนำไปใช้งาน, ค่าของ Cluster ถัดไป
0xFF0 – 0xFF6	0xFFF0 – 0xFFF6	0x?FFFFFF0 – 0x?FFFFFF6	ค่าสำรอง
0xFF7	0xFFF7	0x?FFFFFF7	Cluster เสีย
0xFF8 – 0xFFF	0xFFF8 – 0xFFFF	0x?FFFFFF8 – 0x?FFFFFFF	Cluster สุดท้ายของไฟล์

2.5 สถาปัตยกรรมซีพียู ARM7

สถาปัตยกรรมของ ARM7 เป็นซีพียูแบบ RISC ขนาด 32 บิต ภายในมี Bus ขนาด 32 บิต ตัวเดียวที่ใช้สำหรับรับส่งข้อมูลและคำสั่ง ชุดคำสั่งจะมีขนาด 32 บิตคงที่ ในขณะที่ข้อมูลสามารถเลือกได้ว่าจะมีขนาด 8, 16 หรือ 32 บิต โดยแสดงแกนกลาง (Core) ของ ซีพียู ARM7 ได้ดังรูป



รูปที่ 2.9 แกนกลางของซีพียู ARM7

โครงสร้างของ ARM7 จะเป็นแบบที่เรียบง่าย มีชุดคำสั่งไม่มากนัก ประหยัดพื้นที่สารกึ่งตัวนำที่ใช้สร้าง ประหยัดพลังงาน

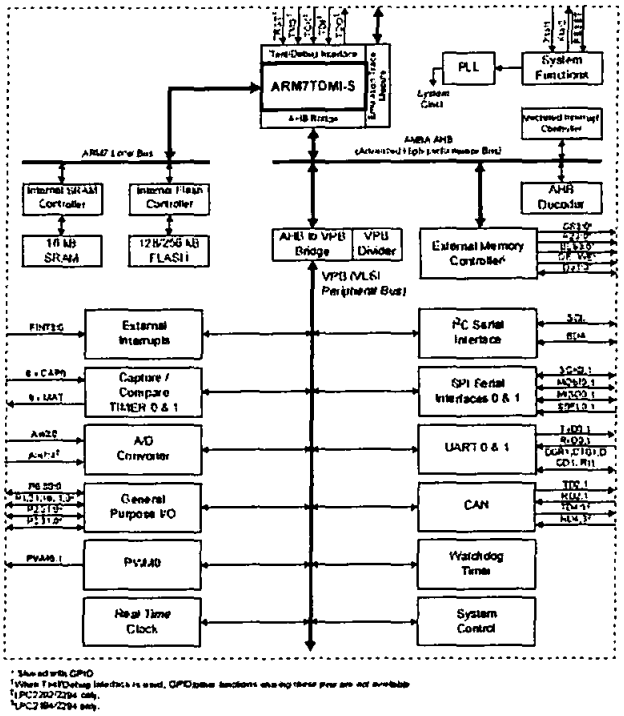
สถาปัตยกรรมของ ARM7 จะเป็นแบบ load-and-store ในการประมวลผลข้อมูลใดๆ ต้องกระทำผ่านทาง Register เริ่มต้นด้วยการโหลดค่าจากหน่วยความจำเก็บใน Register นำค่ามาประมวลผล เสร็จแล้วจะเขียนค่าเก็บในหน่วยความจำดังเดิม

Register ของ ARM7 ที่ใช้งานได้สำหรับผู้ใช้มีทั้งหมด 16 ตัว คือ R0 – R15 โดยทุกตัวมีขนาด 32 บิต โดย R0 – R12 เป็น Register ทั่วไปที่ไม่ได้กำหนดหน้าที่การทำงานพิเศษ ส่วน R13 ทำหน้าที่เป็น Stack Pointer (SP) R14 ทำหน้าที่เป็น Link Register (LR) และ R15 ทำหน้าที่เป็น Program Counter (PC)

2.5.1 ไมโครคอนโทรลเลอร์ ARM 7 Philips LPC2119

- ไมโครคอนโทรลเลอร์ขนาด 16/32 บิต ARM7TDMI-S ในตัวถัง LQFP 64 ขา
- หน่วยความจำ Static RAM ขนาด 16 kB
- หน่วยความจำ Flash Program Memory ขนาด 128 kB อยู่ภายในชิปที่สามารถ ลบ/เขียน ซ้ำได้ถึง 10,000 ครั้ง
- โปรแกรมชิปได้ทันทีผ่าน In System Programming (ISP) และ In-Application Programming (IAP) โดยใช้ซอฟต์แวร์ boot-loader ที่อยู่ภายในชิป
- วงจรแปลงแอนะลอกเป็นดิจิตอลความละเอียด 10 บิต จำนวน 4 ชุด โดยมีเวลาในการแปลงค่าต่ำถึง 2.44 ms
- วงจรไทเมอร์ขนาด 32 บิต 2 ชุด (มี 4 capture และ 4 compare channel)
- PWM (Pulse width modulation) 6 เอาต์พุต
- โมดูลนาฬิกาเวลาจริง (Real Time Clock) และ วอชดอกซ์ (Watchdog)
- วงจรสื่อสารอนุกรม UART (16C550) จำนวน 2 ชุด
- วงจรสื่อสารอนุกรม I^2C ความเร็วสูง (400 kbits/s)
- วงจรสื่อสารอนุกรม SPI 2 ชุด
- มีวงจร Phase lock loop ภายในเพื่อคูณค่าให้สัญญาณนาฬิกาภายในทำงานที่ความถี่สูงสุด 60 MHz
- Vectored Interrupt Controller ที่สามารถกำหนดลำดับความสำคัญ และกำหนดแอดเดรสของเวกเตอร์ได้
- ใช้กับแหล่งจ่ายไฟชุดเดียวขนาด 3.0 V ถึง 3.6 V (3.3 V ± 10%)
- มี I/O pin อเนกประสงค์ที่สามารถใช้กับระดับแรงดัน 5 V ได้สูงสุด 45 ขา โดยสามารถจัดเป็นขาอินเตอร์รัปต์จากภายนอกได้สูงสุด 12 ขา
- มีโหมดประหยัดพลังงาน 2 โหมด ได้แก่ Idle และ Power-down

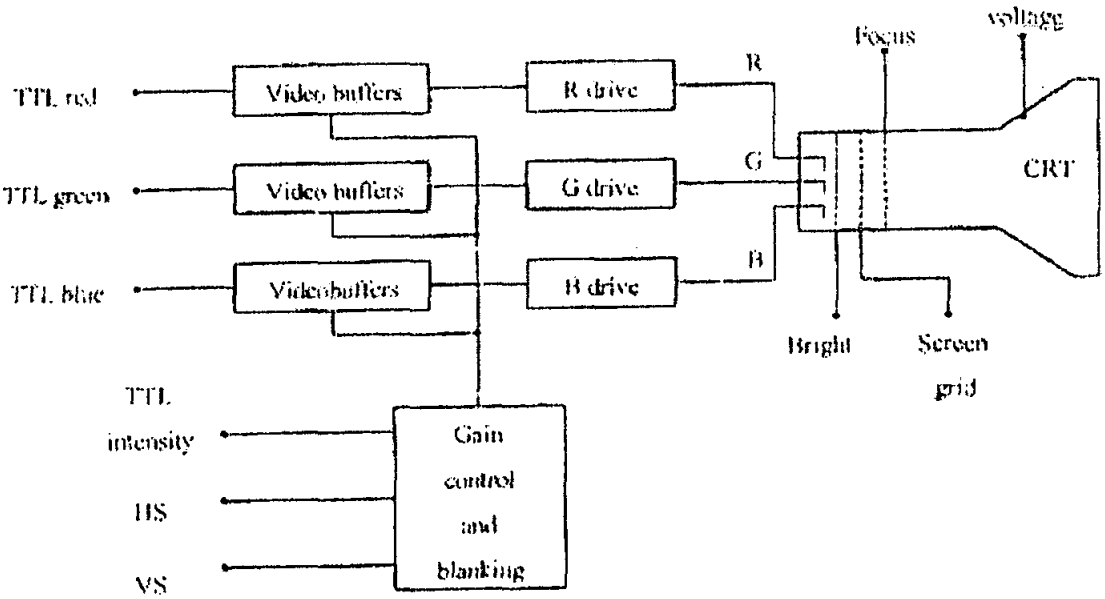
2.5.2 บล็อกไดอะแกรมของ LPC2119



2.6 การสร้างสัญญาณภาพ

2.6.1 การสร้างภาพสีของจอมอนิเตอร์

การสร้างภาพสีของจอมอนิเตอร์เกิดขึ้นได้โดยการใช้หลอดภาพสีที่มีโครงสร้างภายนอกเหมือนกับของหลอดภาพขาวดำ แต่ด้านในของจอภาพจะฉาบเอาไว้ด้วยสารฟอสเฟอร์ 3 ชนิดที่มีคุณสมบัติเปล่งแสงสีแดง สีเขียว และสีน้ำเงินออกมาเมื่อถูกลำอิเล็กตรอนวิ่งมาชน ลักษณะการฉาบสารฟอสเฟอร์จะใช้วิธีการฉาบเป็นจุดหรือเป็นเส้นที่เล็กมากเรียงสลับกันไปจนเต็มตลอดหน้าจอ เนื่องจากสารฟอสเฟอร์มี 3 ชนิด ดังนั้นจึงต้องใช้อิเล็กตรอนกันถึงสามอันเพื่อแยกยิงลำอิเล็กตรอนแต่ละลำให้ไปชนสารฟอสเฟอร์แต่ละสี โดยจะตั้งอิเล็กตรอนกันแต่ละอัน ให้ยิงไปชนแต่ละสีหนึ่งสีใดโดยเฉพาะ เช่นอิเล็กตรอนกันที่ยิงลำอิเล็กตรอนกันไปชนเฉพาะสารฟอสเฟอร์สีแดงจะทำการสร้างแต่สีแดงเท่านั้นให้ปรากฏออกมาหน้าจอภาพทุกครั้งที่อิเล็กตรอนกันอันนี้ทำงานจึงเรียกว่าอิเล็กตรอนกันสีแดง ส่วนอิเล็กตรอนกันที่เหลืออีก 2 อัน ซึ่งตั้งเล็งยิงลำอิเล็กตรอนไปชนสารฟอสเฟอร์สีเขียวและสีน้ำเงิน เพื่อสร้างสีเขียวและสีน้ำเงินให้เกิดที่หน้าจอเราเรียกว่า อิเล็กตรอนกันสีเขียว และอิเล็กตรอนกันสีน้ำเงินหลอดภาพของจอมอนิเตอร์ ก็จะสร้างแสงให้เกิดขึ้นที่หน้าจอโดยมีวงจรไบอัสจ่ายแรงดันให้กับชิ้นส่วนต่างๆ ที่อยู่ในอิเล็กตรอนกันทั้ง 3 เพื่อให้ยิงลำอิเล็กตรอนไปชนจอโดยปรับระดับไบอัสเพื่อทำให้ปริมาณอิเล็กตรอนทั้ง 3 ที่ยิงไปชนจอพร้อมๆ กันนั้นสร้างแสงขาวให้ปรากฏขึ้นที่หน้าจอ โดยมีความสว่าง 50% ขณะที่ยังไม่มีสัญญาณป้อนเข้ามายังอิเล็กตรอนกัน ลำอิเล็กตรอนทั้งสามจะถูกเบี่ยงเบนโดยสนามแม่เหล็กจากขดลวดชุดเบี่ยงเบนทางแนวนอนและแนวตั้งเพื่อให้เกิดการกราดสร้างแสงขึ้นเต็มจอ สัญญาณที่ส่งมาที่อิเล็กตรอนกันทั้งสามต้องมีสามสัญญาณเช่นกัน คือ สัญญาณสีแดงส่งป้อนให้อิเล็กตรอนกันสีแดง เพื่อให้เกิดการสร้างแสงสีแดง และสัญญาณสีเขียวกับสีน้ำเงินส่งป้อนให้กับอิเล็กตรอนกันสีเขียวกับอิเล็กตรอนกันสีน้ำเงิน



รูปที่ 2.10 การป้อนสัญญาณขับจอมอนิเตอร์

เพื่อสร้างแสงสีเขียวและสีน้ำเงิน สัญญาณทั้ง 3 จะป้อนเข้าไปเพื่อทำให้อิเล็กตรอนกันแต่ละอันเปลี่ยนแปลงปริมาณของลำอิเล็กตรอน ที่ยิงไปชนจอ เป็นการสร้างสีต่างๆ ให้เกิดบนจอตามต้องการ เช่น เมื่อส่วนของสัญญาณที่ส่งมาเป็นสีแดงอิเล็กตรอนกันสีแดงจะได้รับสัญญาณเพื่อเพิ่มปริมาณอิเล็กตรอนที่ยิงไปชนสารฟอสเฟอร์สีแดงมากขึ้นจึงเกิดการเปล่งแสงเป็นสีแดง โดยอิเล็กตรอนกัน 2 อันที่เหลือจะถูกทำให้เกิดคัทออฟถ้าส่วนของภาพเป็นสีไซอันก็จะมีสัญญาณสีเขียวและสีน้ำเงินป้อนให้อิเล็กตรอนกันสีเขียวและสีน้ำเงินเพื่อทำให้เพิ่มปริมาณลำอิเล็กตรอนที่ยิงไปชนสารฟอสเฟอร์ที่หน้าจอทำให้เกิดการเปล่งแสงสีเขียวและสีน้ำเงินออกมาพร้อมๆ กันซึ่งทำให้เกิดผลรวมเป็นสีไซอัน โดยอิเล็กตรอนกันสีแดงจะทำให้คัทออฟในจังหวะนั้นสำหรับการสร้างภาพขาวดำนั้นหลอดภาพจะได้รับสัญญาณพร้อมกันทั้ง 3 สัญญาณทำให้อิเล็กตรอนกันทั้งสาม เปลี่ยนแปลงลำอิเล็กตรอนที่ยิงไปชนจออย่างเป็นสัดส่วนต่อกันจึงเกิดการสร้างส่วนของภาพที่เป็น สีขาว เทาอ่อน เทาแก่ตามลำดับ และดำ ที่หน้าจอเราเรียกสัญญาณขาวดำว่า สัญญาณลูมิแนนซ์ (Luminance) หรือเรียกย่อๆ ว่าสัญญาณวาย (Y) คือสัญญาณความสว่าง วิธีการสร้างสัญญาณ Y ก็คือการนำสัญญาณ R, G, B มารวมกันทางไฟฟ้าตามสัดส่วนรู้สึกสว่างเทียบกับแสงสีขาว (15) ดังสมการต่อไปนี้ $Y = 0.3R + 0.59GB + 0.11B$

สำหรับสีขาว มีส่วนผสมแม่สีคือ R =1, B = 1, G = 1 แทนในสมการจะได้

$Y = 0.3 + 0.59 + 0.11$

$Y = 1$ ได้สีขาวสว่าง 100%

สำหรับสีแดง มีส่วนผสมของแม่สีคือ $R = 1, G = 0, B = 0$ แทนในสมการจะได้

$Y = 0.3$ ได้สีแดงความสว่าง 30%

สำหรับสีเขียว มีส่วนผสมแม่สีคือ $R = 0, G = 1, B = 0$ แทนในสมการจะได้

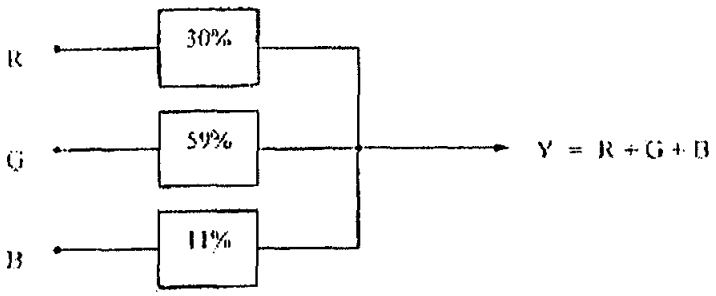
$Y = 0.59$ ได้สีเขียวความสว่าง 59%

สำหรับสีน้ำเงิน มีส่วนผสมแม่สีคือ R = 0, G = 0, B = 1 แทนในสมการจะได้

Y = 0.11 ได้สีน้ำเงินความสว่าง 11%

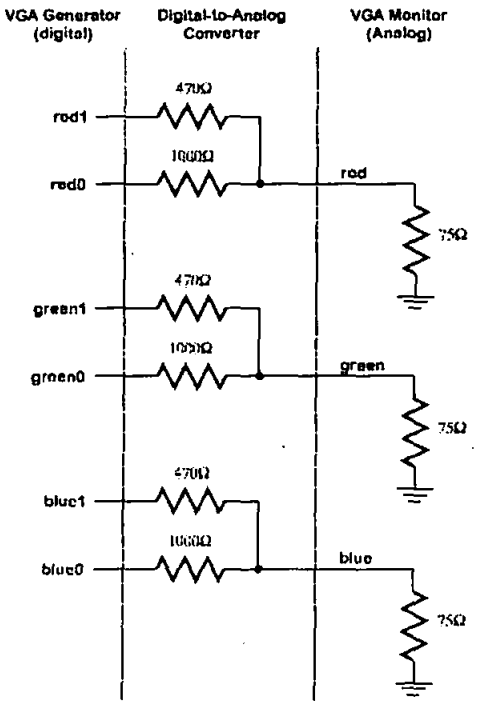
เราสามารถสร้างสัญญาณ Y ได้จากสัญญาณ R, G, B โดยใช้ตัวต้านทานลดระดับแรงดันของสัญญาณทั้ง 3 ให้ได้ขนาดความแรงของสัญญาณ เป็นสัดส่วนตามสมการที่กล่าวมาแล้วคือ

Y = 0.3R + 0.59G + 0.11B แล้วนำสัญญาณที่ได้มารวมกันดังรูป



รูปที่ 2.11 การสร้างสัญญาณลูมิแนนซ์

การสร้างสัญญาณสี 3 สี ประกอบด้วย สีแดง สีเขียว และสีน้ำเงิน โดยจะส่งข้อมูลสีไปยังหน้าจอมอนิเตอร์ โดยสัญญาณแต่ละสัญญาณจะนำไปขับปืนลำแสงอิเล็กตรอนโดยจะยิงออกไป และจะเป็นจุดสีที่หน้าจอมอนิเตอร์ ระดับสัญญาณ Analog มีค่าระหว่าง 0 ถึง 0.7 V ในการควบคุมเส้น ทำให้สีรวมกันเป็น Dot (หรือ Pixel) บนหน้าจอมอนิเตอร์ โดยสีที่เป็นอินพุท Analog สามารถตั้งค่าตั้งแต่ระดับ 1 ถึงระดับ 4 โดยค่าเอาท์พุทที่เป็น Digital 2 ค่า ดังรูปที่ 2.12 ค่าของ 4 ระดับที่อาจเป็นไปได้ในการรวมค่าอินพุทของ Analog โดยมอนิเตอร์สร้าง Pixel = 4x4x4 = 64 สีที่ต่างกัน ดังนั้นค่า Digital 6 ค่าจะสามารถควบคุมเส้นจากสีที่รวมกัน 64 สี

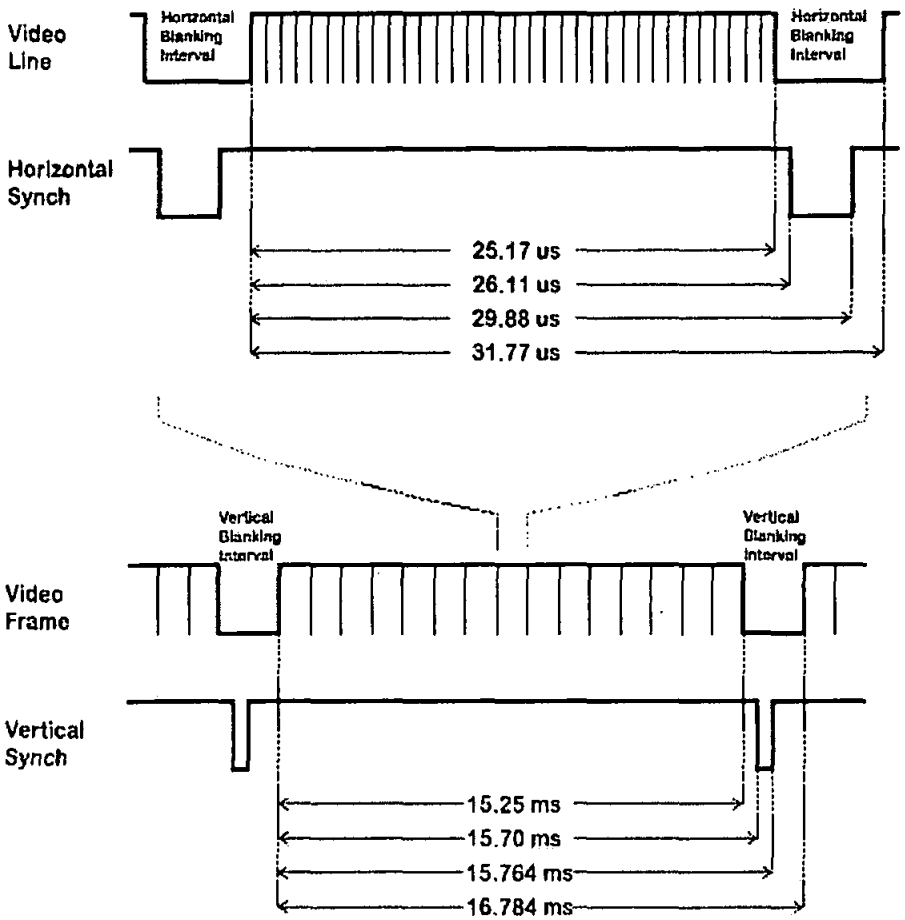


รูปที่ 2.12 แสดงการแปลงค่าเอาท์พุทแบบดิจิตอล ไปเป็นอนาลอก

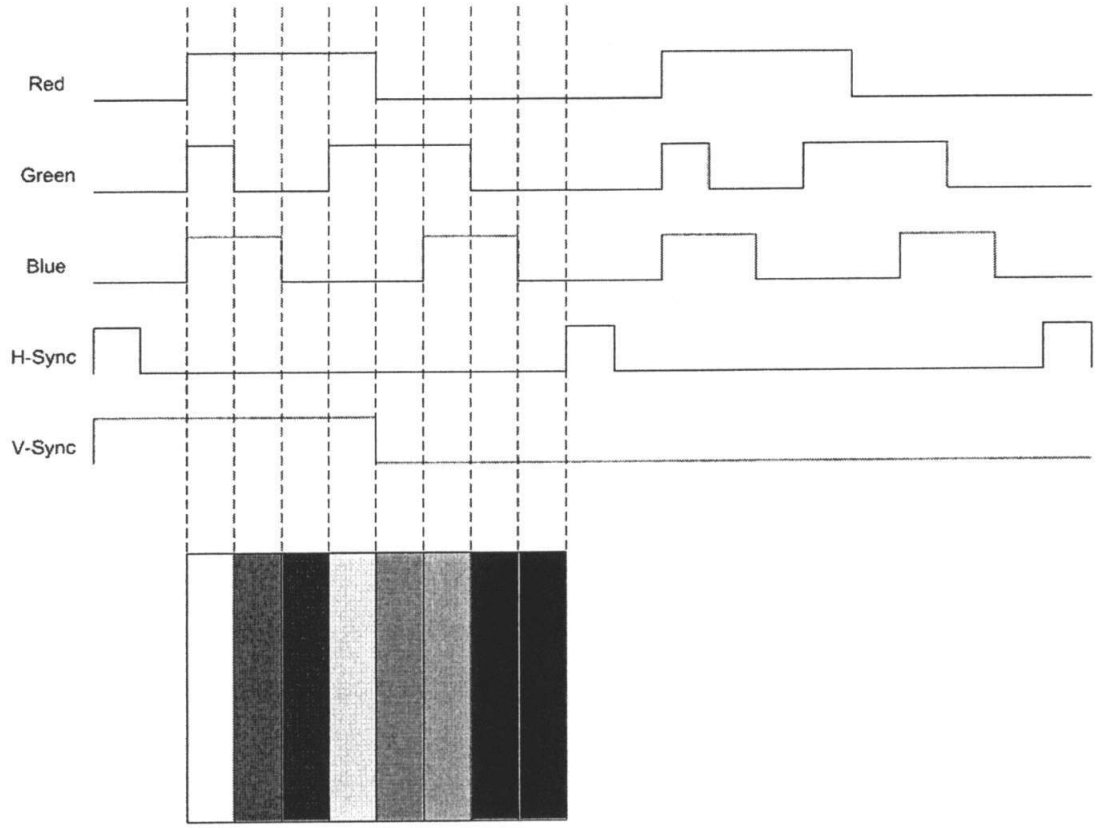
จุดสี (Dot) บนจอมอนิเตอร์ ไม่สามารถให้ข้อมูลมากๆ ได้และเส้นตามแนวนอนของพิกเซลให้ข้อมูลน้อยมากๆ แต่เส้นขอบหลายเส้นสามารถแสดงภาพบนจอมอนิเตอร์ได้ เส้นขอบของสัญญาณ VGA จะมีเส้น 480 เส้น แต่ละเส้นจะมี 640 พิกเซล โดยวงจรในมอนิเตอร์จะมีปืนยิงลำแสงอิเล็กตรอนเพื่อจะส่งอิเล็กตรอนจากไปยังจอมอนิเตอร์ ปืนคู่นี้จะยิงลำแสงจากซ้ายไปขวาและบนลงล่าง ตัดผ่านบนจอภาพจะมีสัญญาณตรงกัน (synchronization) ในคำสั่ง เริ่มและหยุดที่เวลาจริง ดังนั้นเส้นของพิกเซลที่ตัดกันบนจอมอนิเตอร์และเส้นจากบนลงล่างจากรูปภาพ จะมีสัญญาณเวลาตรงกันดังรูปที่ 2.13

สัญญาณแนวนอนที่ Synchronization จะเริ่มต้นที่เส้นและพิกเซลของจอมอนิเตอร์ระหว่างซ้ายและขวา พิกเซลที่ส่งไปยังจอมอนิเตอร์ใช้เวลาไม่เกิน 25.17 us สัญญาณในการ Synchronization ที่ลดลงต่ำสุดประมาณ 0.94 us หลังจากพิกเซลสุดท้าย และยังคงต่ำที่ 3.77 us เส้นพิกเซลใหม่สามารถเริ่มที่ค่าต่ำๆ คือ 1.89 us หลังจากเส้นแนวนอนหยุดนิ่ง ดังนั้นเส้นสัญญาณจะอยู่ระหว่างค่า 25.17 us ถึง 31.77 us
สัญญาณแนวตั้งที่ Synchronization จะเริ่มต้นที่เส้นและพิกเซลของจอมอนิเตอร์ระหว่างบนและล่างของมอนิเตอร์ เส้นจะส่งสัญญาณไปยังจอมอนิเตอร์ด้วยเวลา 15.25 ms

สัญญาณในการ Synchronization ที่ลดลงต่ำสุดประมาณ 0.45 ms หลังจากพิกเซลสุดท้ายและยังคงต่ำที่ 64 us เส้นแรกที่ถัดจากขอบเริ่มต้นด้วยค่าน้อย 1.02 ms หลังจากเส้นแนวตั้ง sync สัญญาณจากขอบมีค่าระหว่าง 15.25 ms ถึง 16.784 ms



รูปที่ 2.13 แสดงสัญญาณเวลาที่ synchronous กันของเส้นแนวนอนและแนวตั้ง

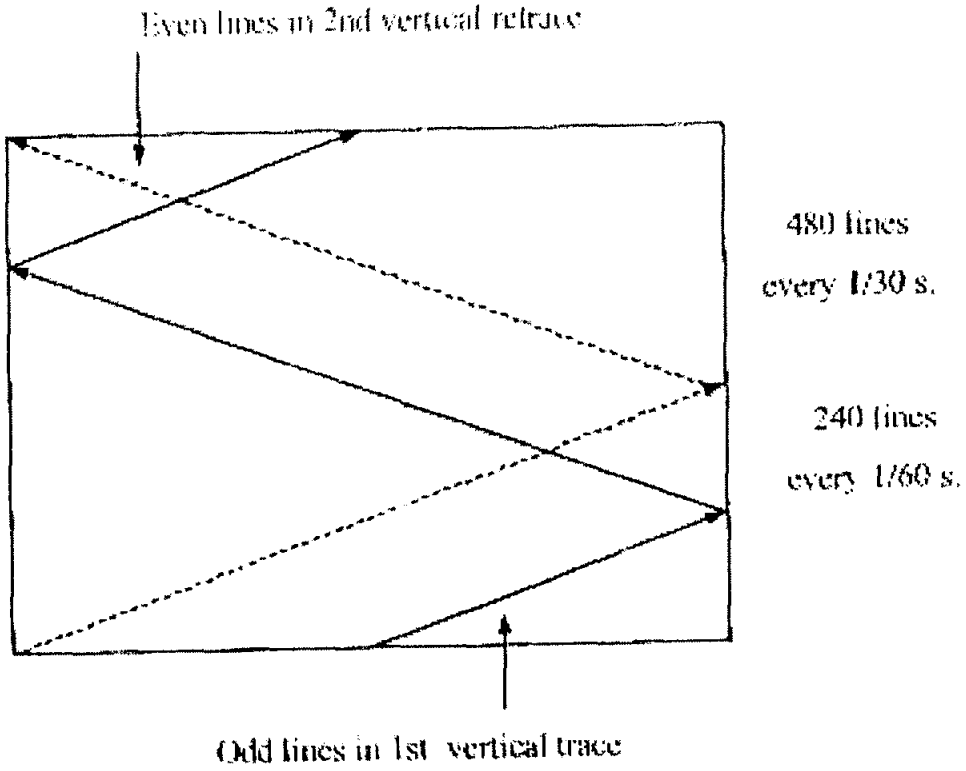


รูปที่ 2.14 ตัวอย่างการสแกนภาพ

2.6.2 การสแกน

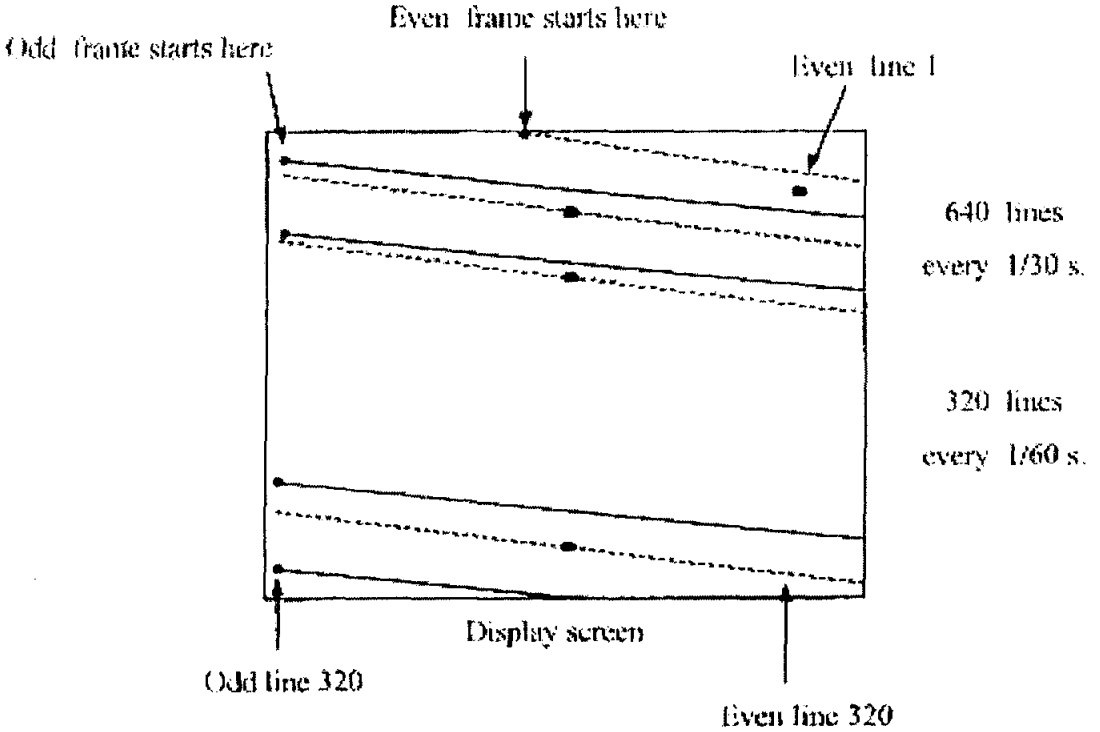
การสแกนคือการนำเอาสัญญาณภาพที่อยู่ในรูปของของสัญญาณไฟฟ้านำมาเรียงกันให้เกิดเป็นภาพ โดยการกวาดเป็นเส้นภาพที่หน้าจอโดยตัวที่มีหน้าที่สำคัญคือหลอดภาพ หลอดภาพมีโครงสร้างคล้ายกับกับหลอดสุญญากาศทั่ว ๆ ไปที่ปล่อยอิเล็กตรอนมาจาก Cathode แล้วมีการดึงลำอิเล็กตรอนให้วิ่งเป็นลำกระทบเข้ากับหน้าจอหรือ Anode การสแกนมี 2 วิธีคือ การสแกนแบบเดินหน้า (Progressive Scanning) กับการสแกนแบบสลับเส้น (Interlaced Scanning)

การที่จะทำให้การสแกนมีความต่อเนื่องขององค์ประกอบภาพต้องคำนึงถึงหลัก 3 ประการ คือ ลำอิเล็กตรอนที่กวาดไปทางแนวนอน (Horizontal Scanning) ในแต่ละครั้งจะต้องครอบคลุมองค์ประกอบภาพทั้งหมดของเส้นนั้น ในแต่ละเส้นของการสแกนลำอิเล็กตรอน ลำแสงต้องกวาดกลับด้วยความเร็วสูงไปยังด้านซ้ายเพื่อเริ่มการสแกนในลำดับต่อไป เวลาของการสะบัดกลับเราเรียกว่า Retrace หรือ Fly back ในกรณีดังกล่าวจะต้องไม่มีข้อมูลภาพใดๆ เพราะหลอดภาพจะเกิดการ Blank out ในขณะนั้น ในขณะที่เส้นสแกนสะบัดกลับมาเพื่อเริ่มต้นทางซ้ายใหม่ตำแหน่งทางแนวตั้งต้องต่ำกว่าตำแหน่งเดิมเพื่อทำให้การสแกนเส้นต่อไปไม่ทับกันทั้งนี้โดยการควบคุมทางแนวตั้ง (Vertical Scanning)



รูปที่ 2.15 การสแกนทางแนวนอน

การสแกนที่ใช้ในจอมอนิเตอร์ต้องใช้การเรียงภาพ 30 ถึง 60 ภาพต่อวินาทีจึงจะเกิดเป็นภาพที่ต่อเนื่องแต่ก็ยังมีการกระพริบ (Flicker) เนื่องจากว่าการสแกนเริ่มจากขอบบนลงมาด้านล่างแสงทางด้านบนเริ่มมืดลงกว่าด้านล่างจึงมองเห็นว่ามันกระพริบ และเวลาที่ลำแสงการสแกนวกกลับไปด้านบนด้านล่างก็เกิดปัญหาเช่นเดียวกันความรู้สึกต่อกรณีนี้ก็คือ เกิดแสงกระพริบหรือวูบวาบขึ้นซึ่งจะเกิดขึ้น ในการสแกนแบบเดินหน้า (Progressive Scanning) ซึ่งเป็นการสแกนพื้นฐาน เพื่อแก้ปัญหาการกระพริบจึงต้องใช้การสแกนสลับเส้นหรือการสแกนแบบสอดแทรก (Interlaced Scanning)โดยครั้งแรกจะสแกนที่ฟิลด์คี่ (Odd line trace) และครั้งต่อไปจะสแกนแบบฟิลด์คู่ (Even line trace) เป็นการสแกนแบบเส้นเว้นเส้น หมายความว่าการที่จะได้ภาพ 1 ภาพหรือ 1 เฟรมต้องใช้การสแกนแนวตั้ง 2 ครั้งหรือ 2 ฟิลด์ (Field) มาตรฐานของจอ VGA จะใช้เส้นสแกน320เส้นทางแนวนอนและ 240 เส้นทางแนวตั้งเริ่มต้นการสแกนสมมุติว่าจากเส้นสแกนคี่ทางแนวนอนโดยเริ่มจากทางซ้ายแล้วกวาดไปทางขวานับเป็นเส้น สแกนที่ 1 แล้วจึงสแกนเส้นที่ 3, 5, 7, 9 และต่อๆ ไปจนได้เส้นสแกนครบ 320 เส้นดังแสดงรูปด้านล่าง (ในส่วนที่เป็นเส้นทึบ)

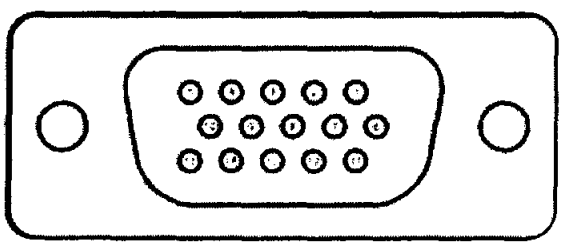


รูปที่ 2.16 การสแกนทางแนวตั้ง

ที่จุดสิ้นสุดการสแกนเส้นคี่ทางแนวนอนนี้จะเป็นจุดเริ่มต้นการสแกนทางแนวตั้งของฟิลด์คี่ซึ่งเราเรียกว่าการสแกนทางแนวตั้งว่า Vertical Retrace หรือ Fly back เพื่อดึงกลับไปยังด้านบนของจอภาพในตำแหน่งนี้เพื่อให้เริ่มต้นการสแกนเส้นคู่ต่อไป ที่กล่าวมาทั้งหมดคือการสแกนในหนึ่งฟิลด์ Retrace Times ทั้งทางแนวนอนและแนวตั้งเป็นเวลาสั้นๆ ถึงอย่างไรก็ตามเราไม่ต้องการให้เส้นสแกนของช่วงที่เป็นการสะบัดกลับเข้ามารบกวนให้เกิดภาพในส่วนนี้จึงต้องทำการลบเส้นสะบัดกลับ Retrace Times จะใช้เวลาประมาณ 10 – 16 เปอร์เซ็นต์ของเวลาทั้งหมดในการสแกน การสแกนของเส้นคู่ก็เป็นในทำนองเดียวกันต่างกันที่จุดเริ่มต้นเท่านั้น โดยจะเริ่มที่จุดสุดท้ายของขั้นตอนก่อนหน้านี้ และก็จะสลับกันเป็นเช่นนี้เรื่อยไป

2.6.3 คอนเนคเตอร์สำหรับ VGA

คอนเนคเตอรสำหรับ VGA เป็นชนิด DB-15 มีขา 3 แถว หรืออาจเรียกว่า D-Sub 15 ใช้กับจอมอนิเตอร์ทั่วไป ซึ่งแสดงแบบขาดังรูป



รูปที่ 2.17 แสดงขาตัวเมียของพอร์ต DB-15

ตารางที่2.9 แสดงตำแหน่งของขา DB-15

ตำแหน่งขา	สัญญาณ	
Pin 1	RED	Red video
Pin 2	GREEN	Green video
Pin 3	BLUE	Blue video
Pin 4	N/C	Not connected
Pin 5	GND	Ground (HSync)
Pin 6	RED_RTN	Red return
Pin 7	GREEN_RTN	Green return
Pin 8	BLUE_RTN	Blue return
Pin 9	+5 V	+5 V (DDC)
Pin 10	GND	Ground (VSync, DDC)
Pin 11	N/C	Not connected
Pin 12	SDA	I^2C data
Pin 13	HSync	Horizontal sync
Pin 14	VSync	Vertical sync
Pin 15	SCL	I^2C clock

บทที่ 3

การออกแบบ

3.1 การติดต่อระหว่างไมโครคอนโทรลเลอร์กับ SD Card

ทางผู้จัดทำได้ใช้วิธีการให้ไมโครคอนโทรลเลอร์ส่งข้อมูลให้กับ SD Card โดยใช้ Port SPI โดยจะให้ไมโครคอนโทรลเลอร์ส่งคำสั่งอ่านข้อมูล 1 บล็อก จาก SD Card ผ่านทาง Port SPI จากนั้นข้อมูลที่ได้จาก SD Card จะถูกส่งกลับมายังไมโครคอนโทรลเลอร์ ซึ่งไมโครคอนโทรลเลอร์จะทำการถอดรหัสไฟล์ภาพ แล้วให้กำเนิดสัญญาณภาพส่งออกทาง Port D-Sub

3.2 วงจรรวมของอุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

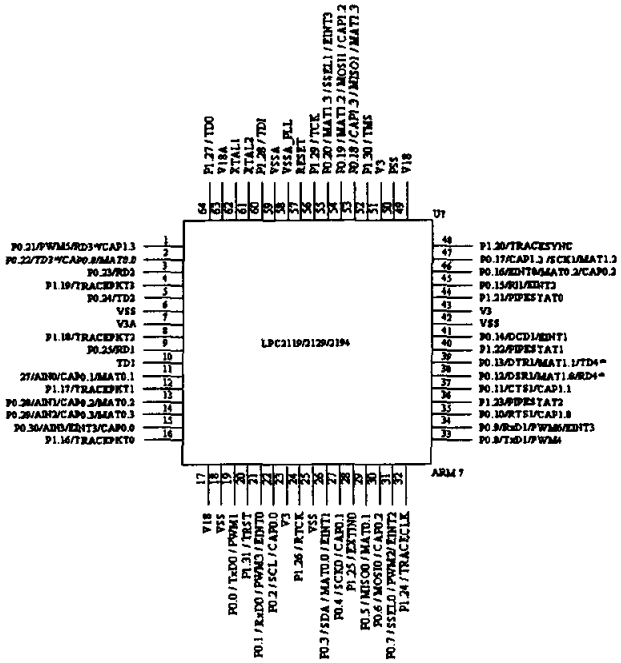
อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์ แบ่งอุปกรณ์เป็น 2 ส่วน ดังนี้

3.2.1 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ที่ใช้เป็นไมโครคอนโทรลเลอร์ตระกูล ARM7 เบอร์ LPC2119 เนื่องจากมีความเร็วในการประมวลผลทางสัญญาณเร็ว มีการรองรับการติดต่อในระบบต่างๆ มากมาย ซึ่งได้เลือกใช้ระบบ SPI Bus ซึ่งมีขนาดของหน่วยความจำที่เพียงพอ ไมโครคอนโทรลเลอร์ตัวนี้จะทำงานที่ 16/32 bits และศึกษาถึงโครงสร้างต่างๆ ในการใช้ไมโครคอนโทรลเลอร์ตัวนี้ ซึ่งเป็นที่นิยมใช้งานในการประมวลสัญญาณที่มีความเร็วสูง

ในส่วนการติดต่อกับ SD/MMC มี 4 ขา คือ SSEL, SCK, SDO, SDI

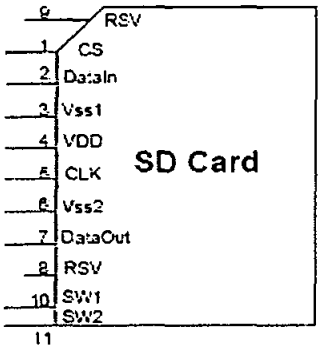
ในส่วนของการสร้างสัญญาณภาพ จะสร้างสัญญาณภาพออกเป็นสัญญาณอนาลอก ออกทาง Port D-Sub



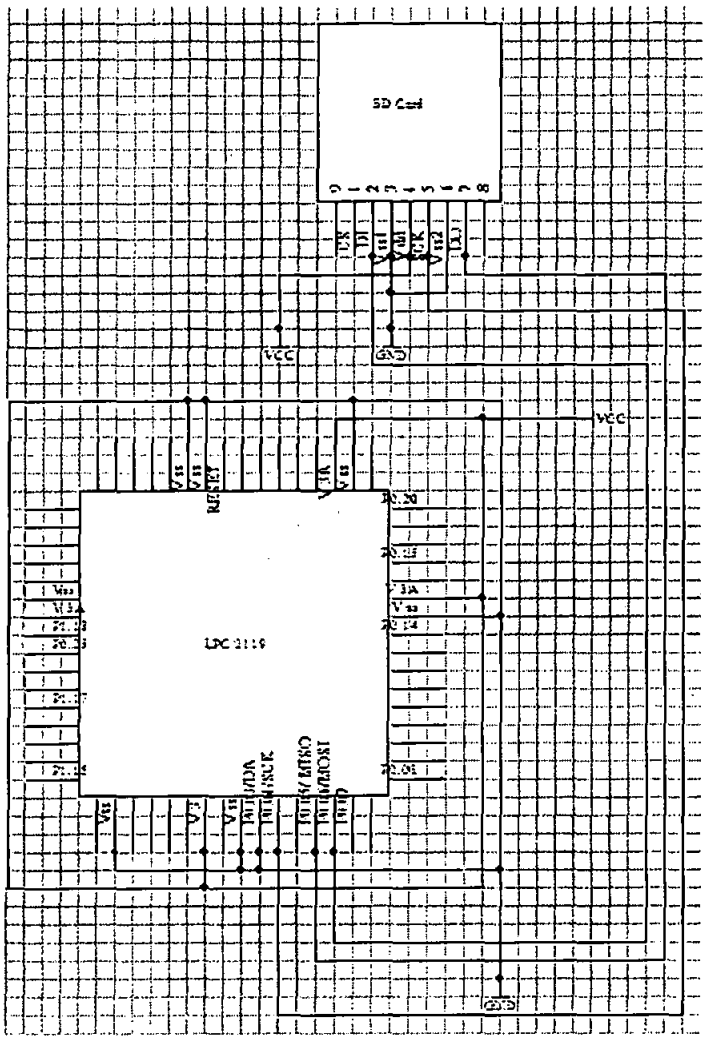
รูปที่ 3.1 ไมโครคอนโทรลเลอร์ตระกูล ARM7 เบอร์ LPC2119

3.2.2 SD Card

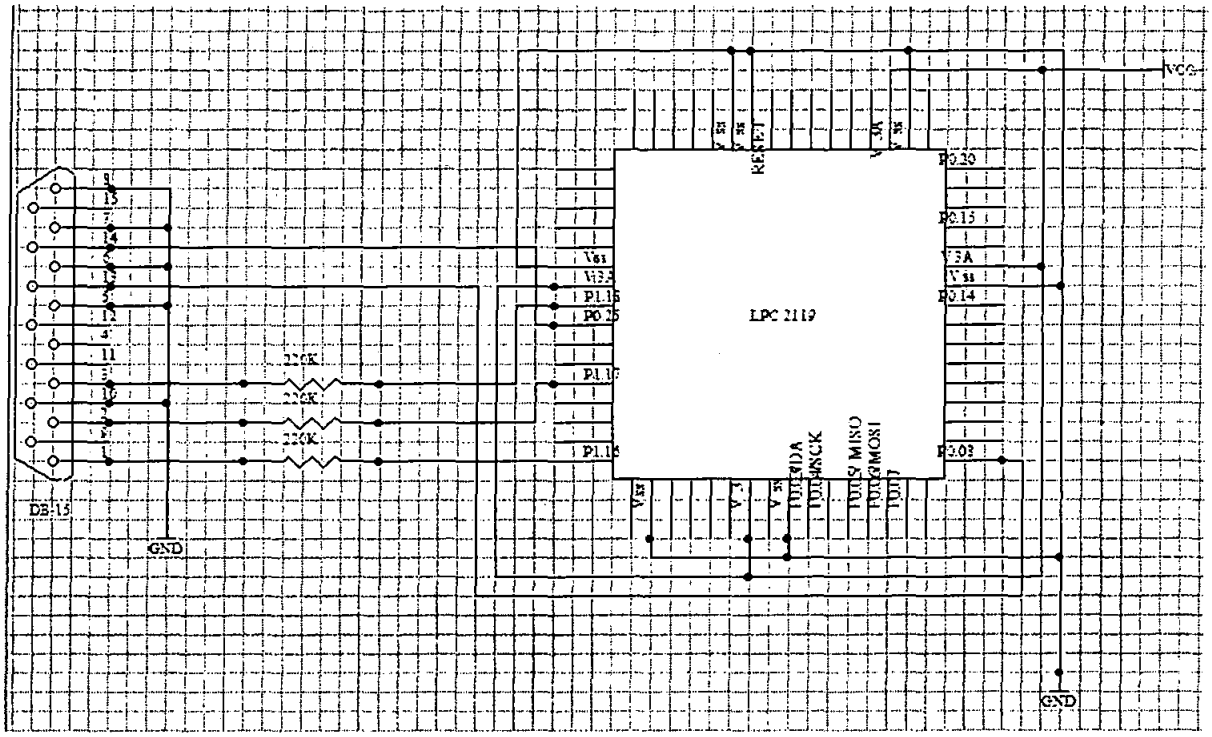
ผู้จัดทำได้เลือกใช้ SD Card ในการเก็บข้อมูลไฟล์ภาพ เนื่องจากเห็นว่ามีความเหมาะสมในเรื่องของราคาที่ไม่ค่อยสูงมาก และเป็น Card ที่หาซื้อได้ง่าย



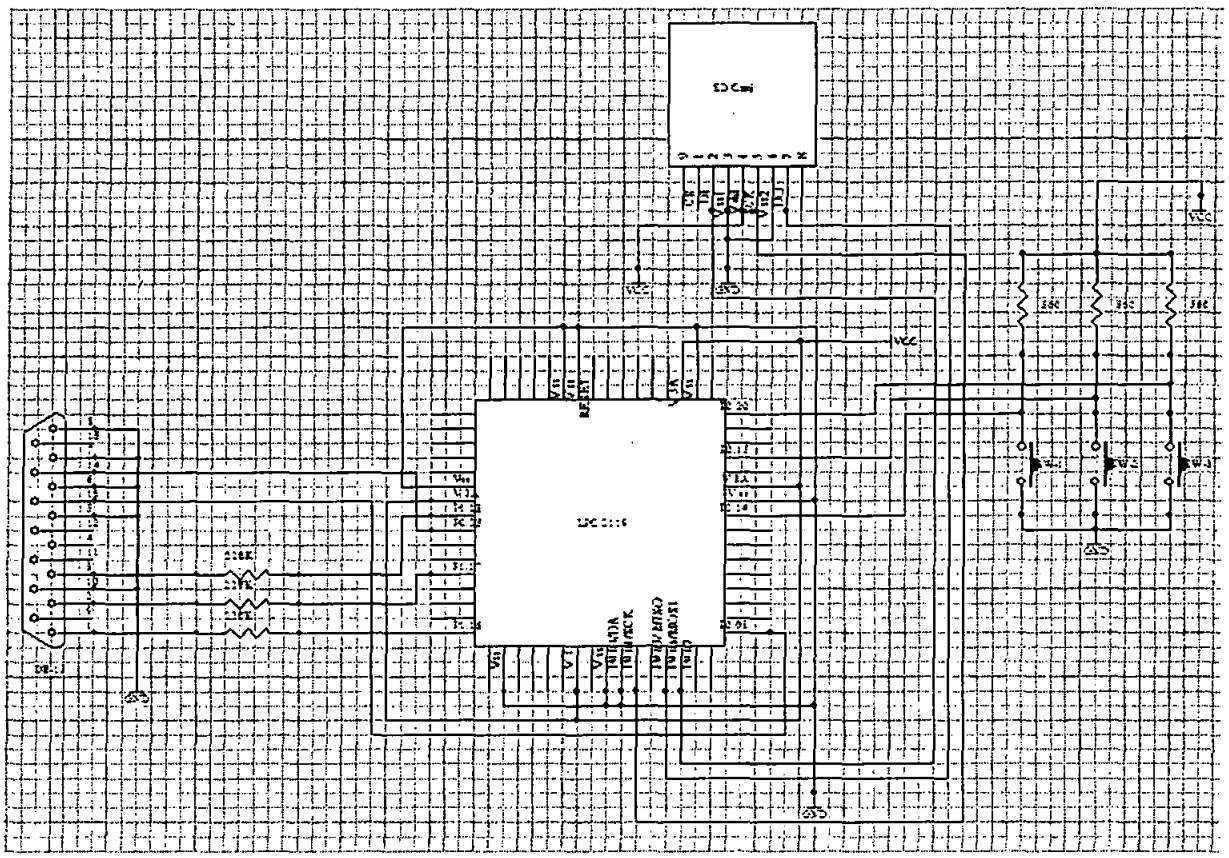
รูปที่ 3.2 SD Card



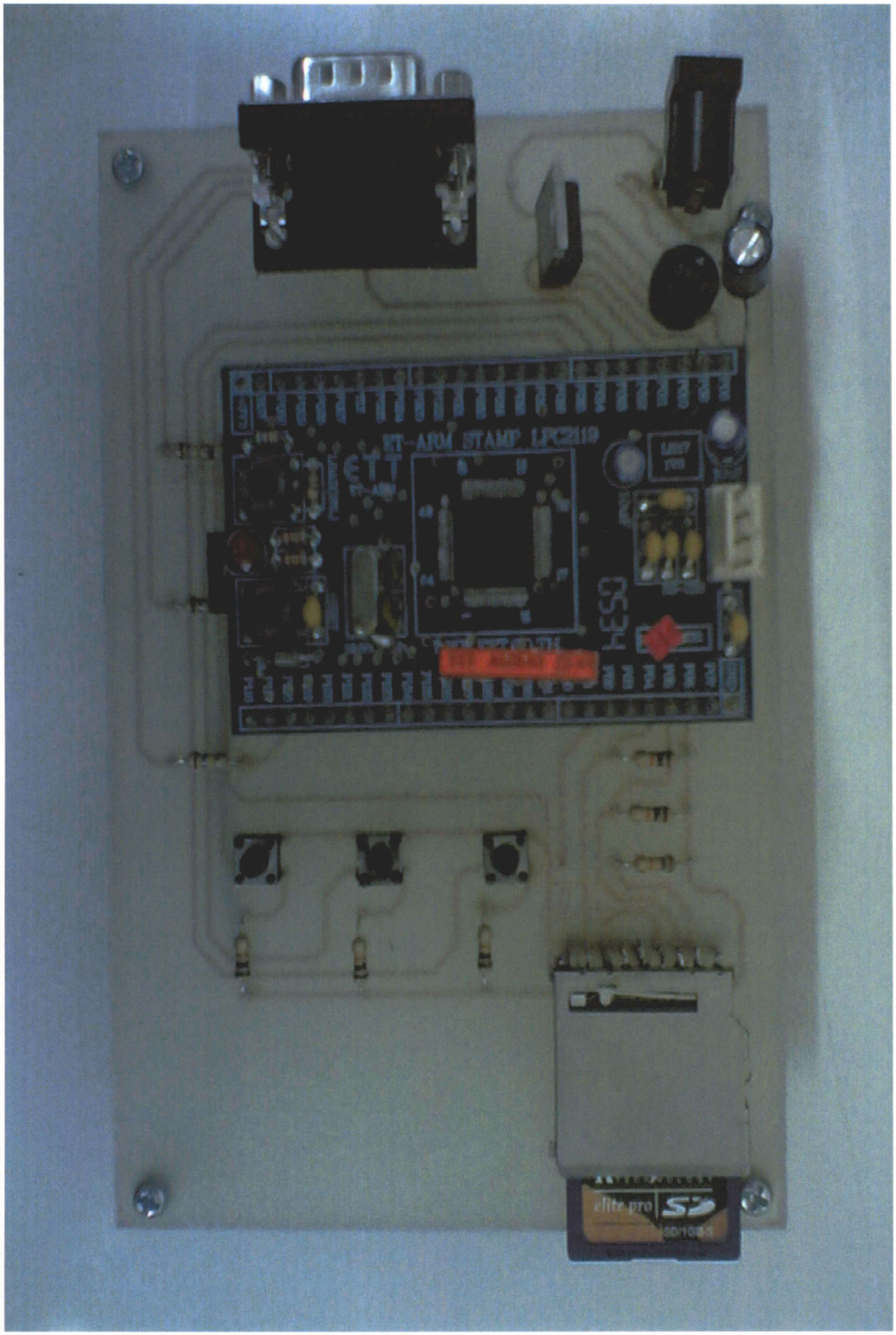
รูปที่ 3.3 วงจรการติดต่อระหว่าง SD Card กับ ARM7 เบอร์ LPC2119



รูปที่ 3.4 วงจรการติดต่อระหว่าง DB-15 กับ ARM7 เบอร์ LPC2119

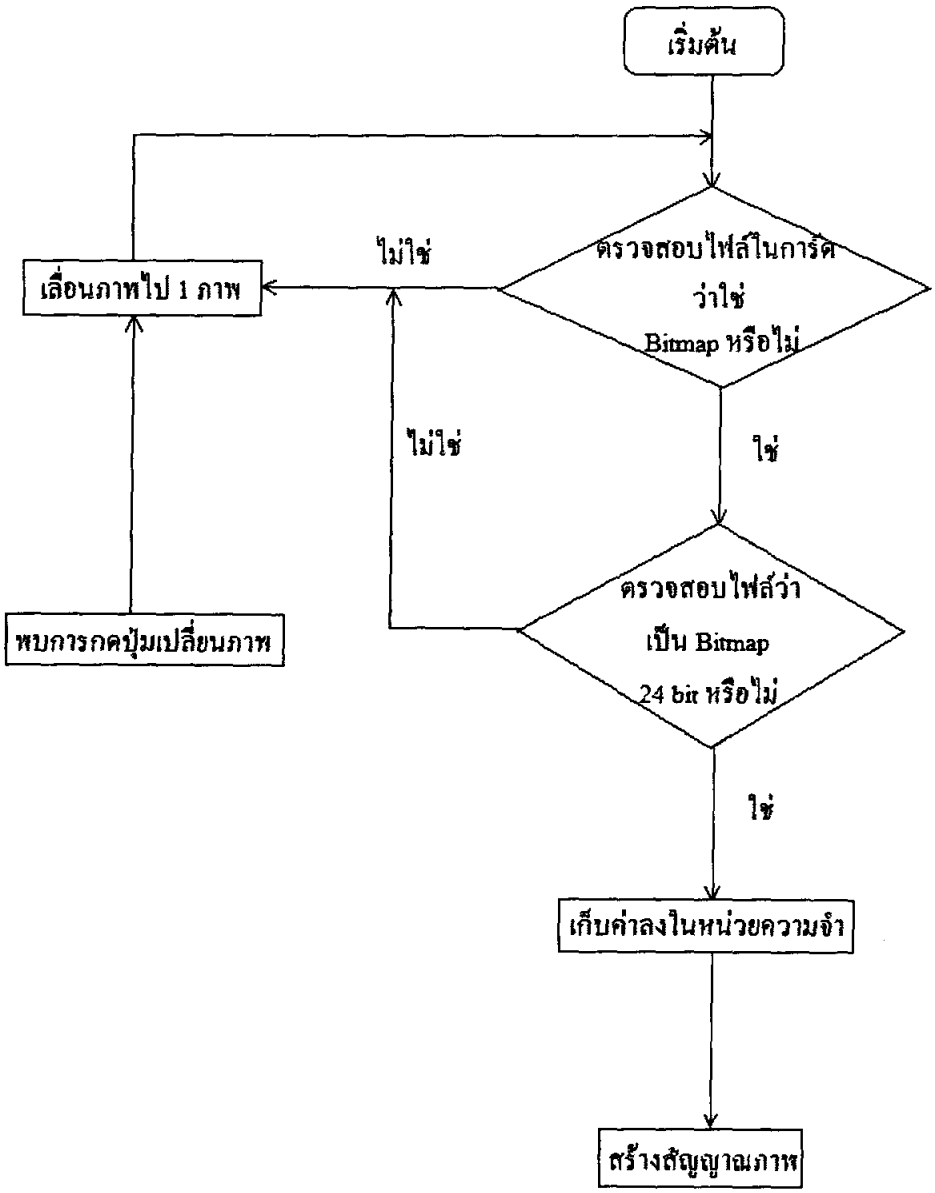


รูปที่ 3.5 วงจรการติดต่อ SD Card, DB-15 และ ARM7 เบอร์ LPC2119



รูปที่ 3.6 อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์

3.3 ผังการทำงานของอุปกรณ์แสดงภาพทางจอมอนิเตอร์



รูปที่ 3.7 ผังการทำงานของอุปกรณ์แสดงภาพทางจอมอนิเตอร์

3.4 การคำนวณ Resolution

การคำนวณหา Resolution ของหน้าจอแสดงผลขนาดมาตรฐาน 640x480 pixel จะใช้เวลาในการสแกนภาพประมาณ 40 us แต่เวลาที่คอนโทรลเลอร์สามารถทำได้คือ 400 us ซึ่งสามารถแสดงการคำนวณได้ดังนี้

จากความละเอียดมาตรฐานของ VGA คือ 640x480 pixel 60Hz เพราะฉะนั้นจะได้ว่า

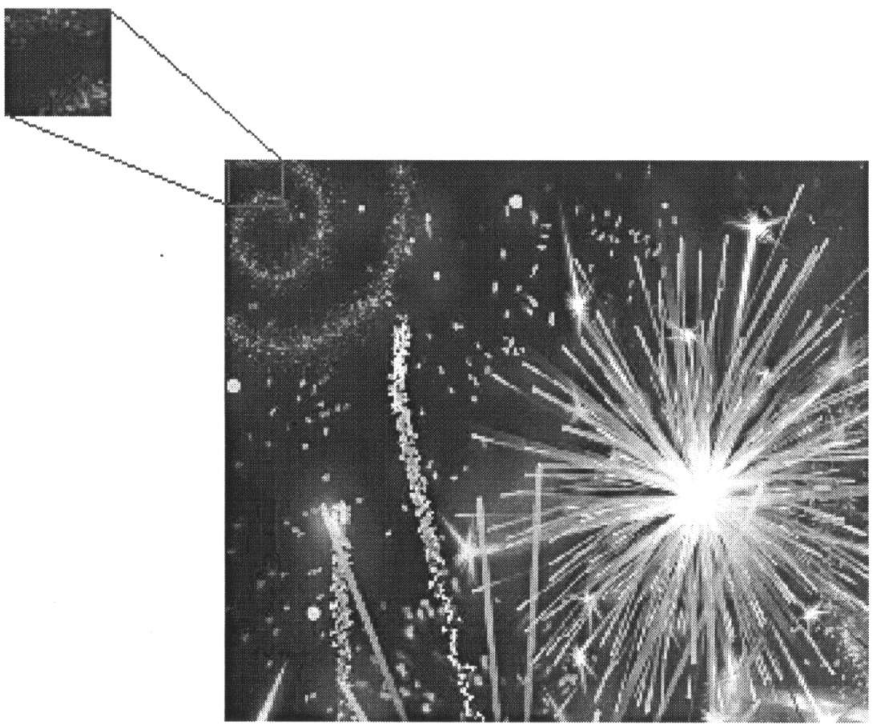
คาบเวลา = $\frac{1}{60}$ = 16.7 ms เพราะฉะนั้น 1 เส้นใช้เวลา = $\frac{16.7\ ms}{480}$ = 34.7 us

ซึ่งจะได้เป็นความถี่ Horizontal Sync = $\frac{1}{34.7\ \text{us}}$ = 29 kHz

เวลาที่สามารถส่งสัญญาณสีออกไปได้เนื่องจากมี front pouch และ back pouch = 34.7 us – 2.5 us = 32.2 us

เพราะฉะนั้น 1 pixel ใช้เวลาในการส่งสัญญาณสี = $\frac{32.2\ us}{640}$ = 50 ns

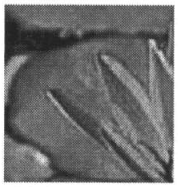
แต่ไมโครสามารถส่งสัญญาณสีได้เร็วสุดแค่ 900 *ns* เพราะฉะนั้นสามารถทำความละเอียดได้สูงสุดได้ = $\frac{32.2\ us}{900\ ns}$ = 35.7 pixel

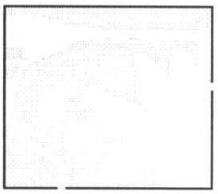


รูปที่ 3.8 แสดงขนาด 40 X40 pixel จากภาพขนาด 640x480 pixels

ไฟล์บิตแมพ 24 Bitจะจัดเรียงแบบ 2 byteโดย2 byte แรกเป็นสีแดง สีเขียว สีน้ำเงินน้ำเงิน

00 00 00 โดย จะแบ่งพิจารณาเป็นสีๆ ไป ถ้าสีไหนมีค่ามากกว่า 0x0F จะตัดให้มีค่าเป็น 1 แต่ถ้าน้อยกว่าจะให้ค่า เป็น 0 แสดงได้ดั้งภาพ





รูปที่ 3.9 เปรียบเทียบระหว่าง 24 bit กับภาพที่แปลงเหลือ 1 bit

บทที่ 4

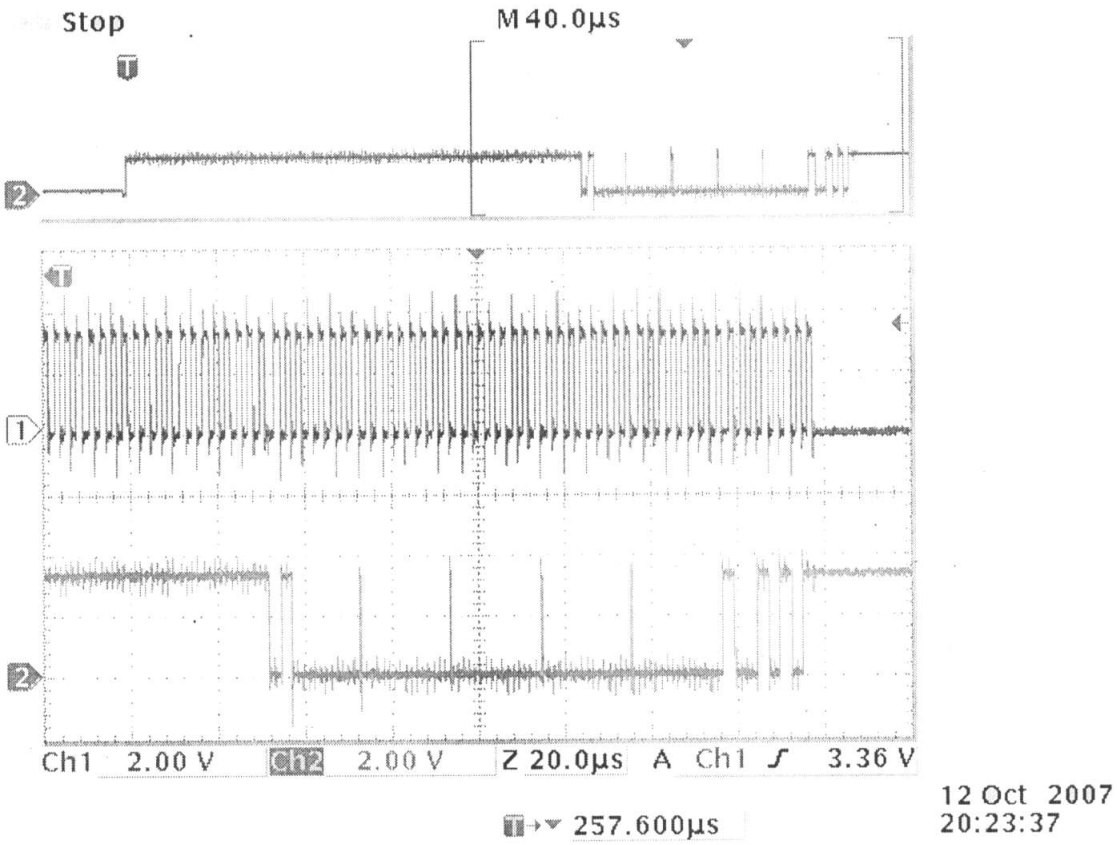
การทดลองและผลการทดลอง

การทดลองแบ่งออกเป็น 2 การทดลอง ดังนี้

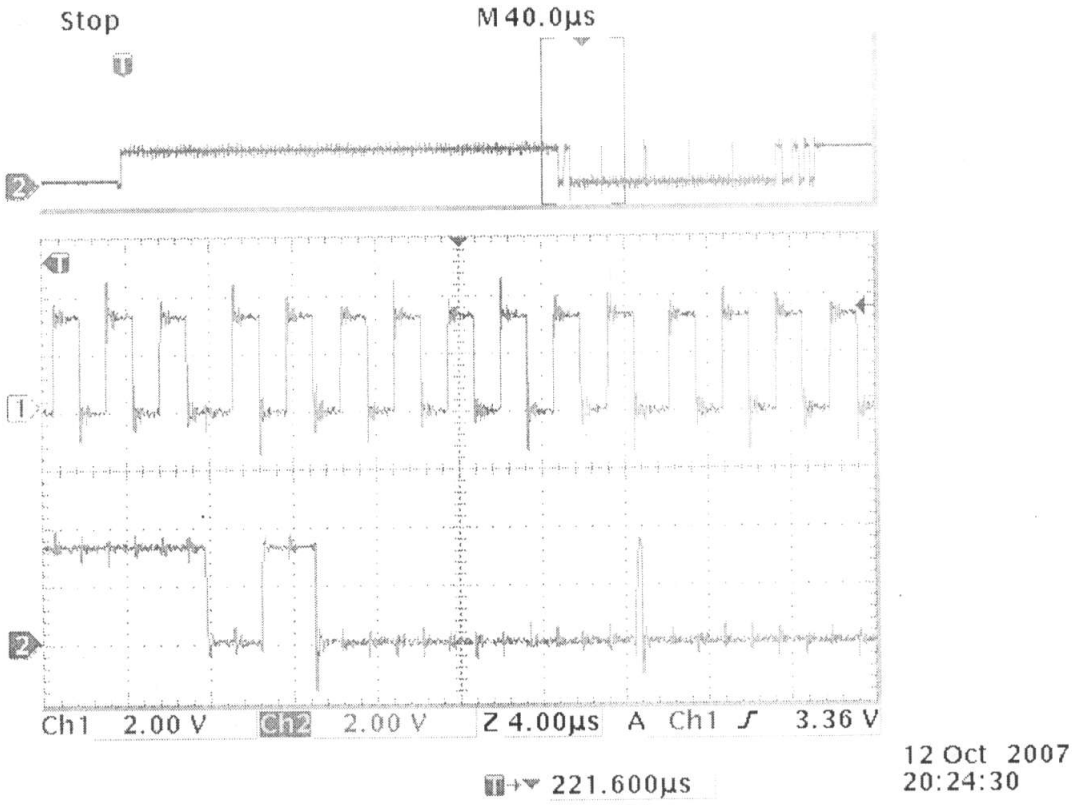
การทดลองที่ 1 วัดสัญญาณของการติดต่อแบบอนุกรมในลักษณะ SPI Bus

การทดลองที่ 2 วัดสัญญาณของหน้าจอมอนิเตอร์

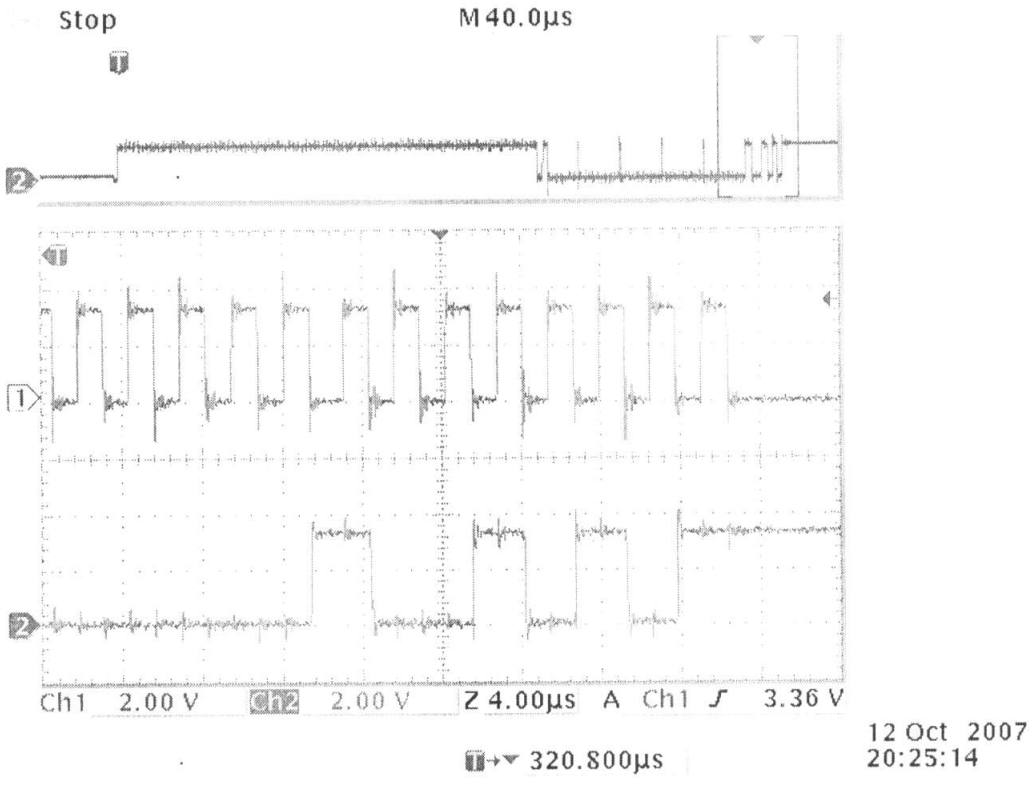
การทดลองที่ 1 วัดสัญญาณของการติดต่อแบบอนุกรมในลักษณะ SPI Bus



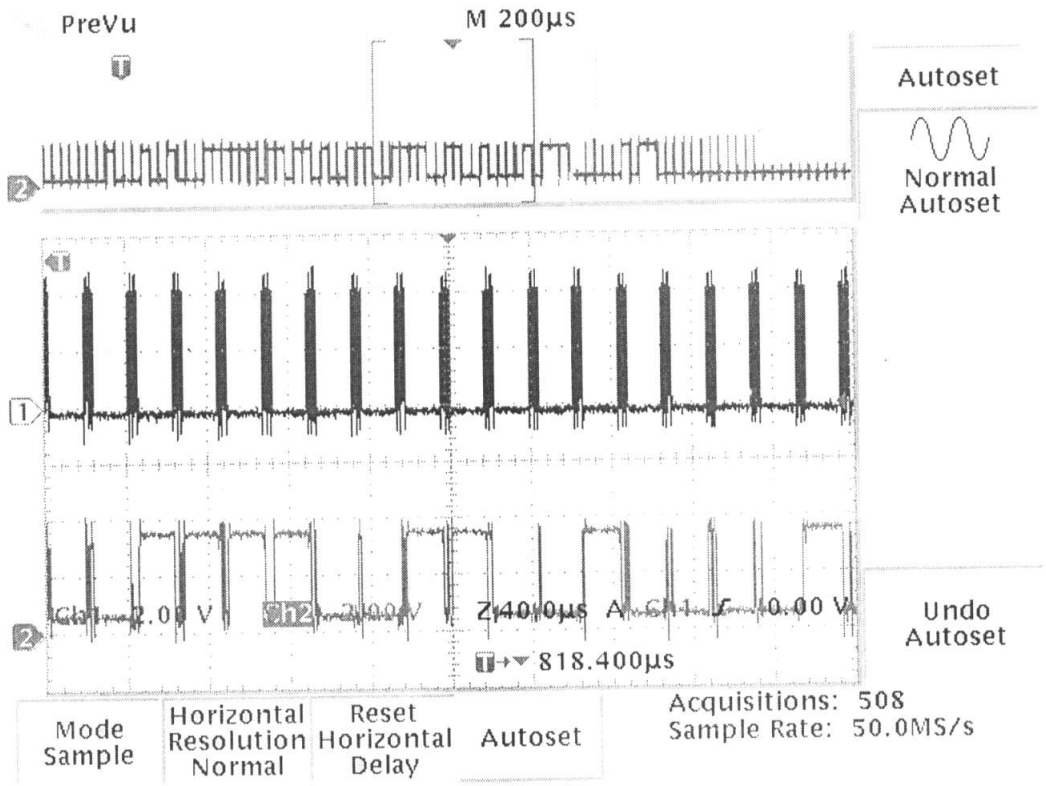
รูปที่ 4.1 รูปสัญญาณของชุดคำสั่งที่ส่งมาจากไมโครคอนโทรลเลอร์



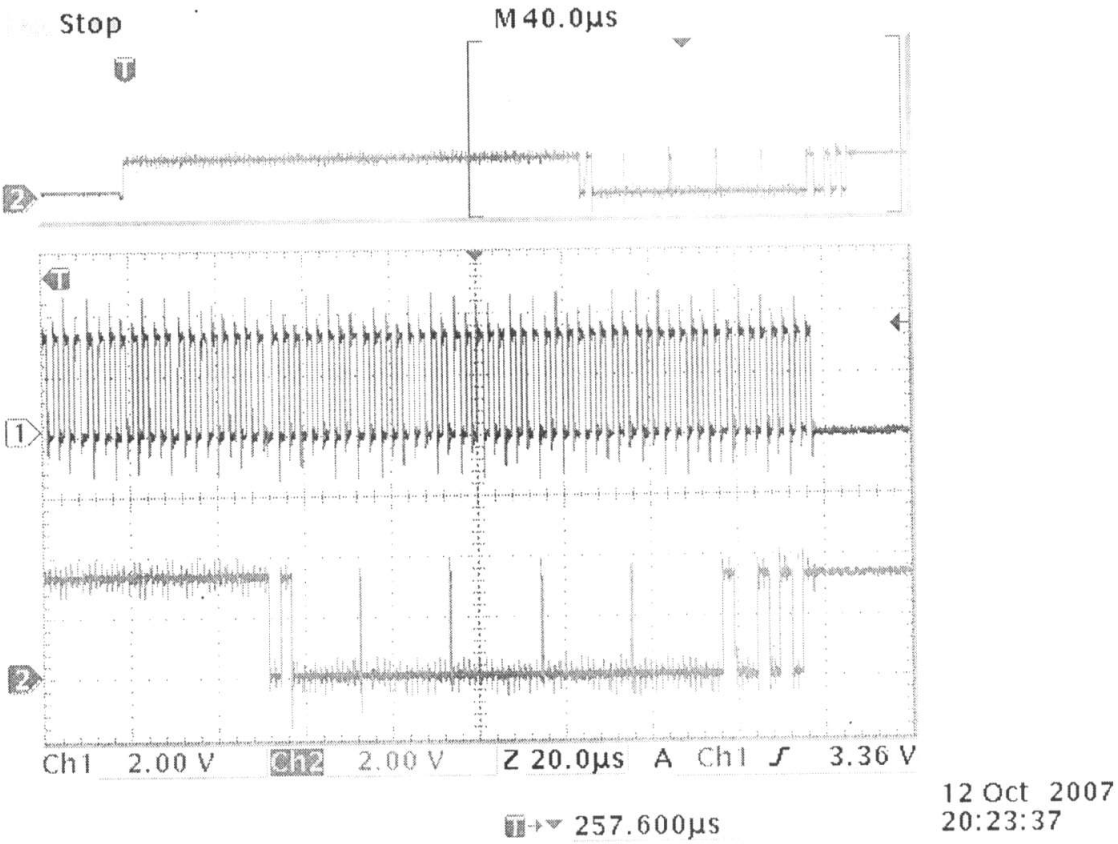
รูปที่ 4.2 รูปสัญญาณคำสั่ง RESET (0x40)



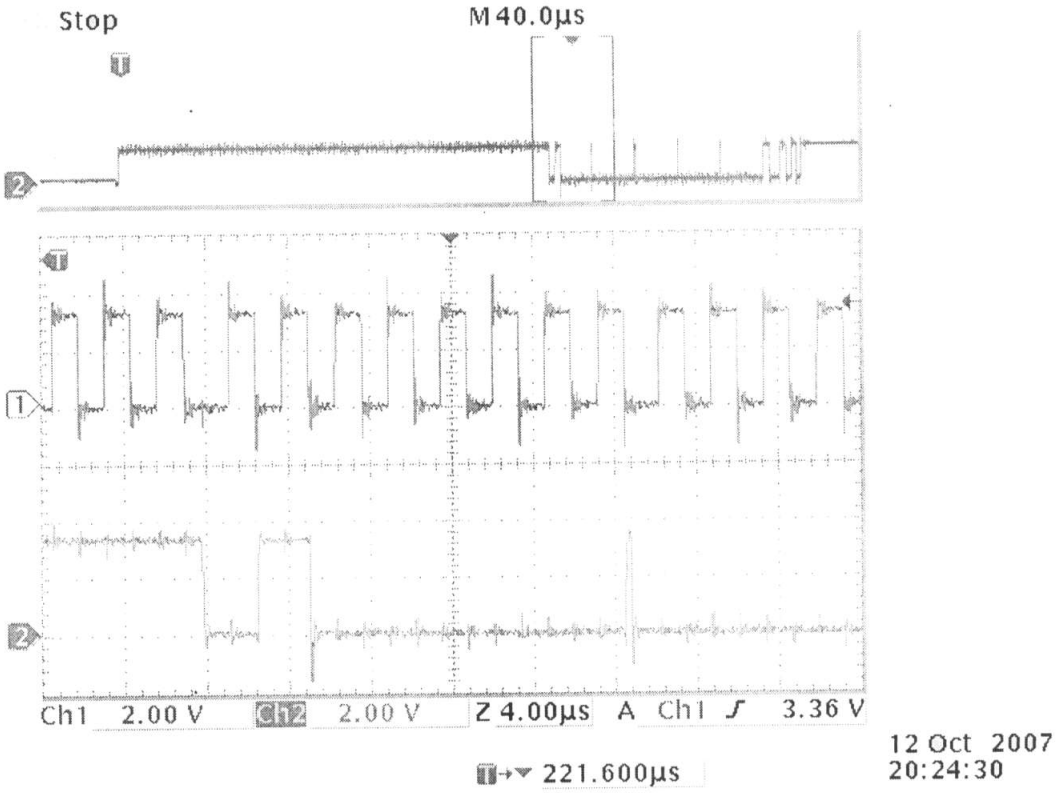
รูปที่ 4.3 รูปสัญญาณคำสั่งRESET (0x95)



รูปที่ 4.4 รูปสัญญาณของชุดข้อมูลที่ได้รับจากการ์ด

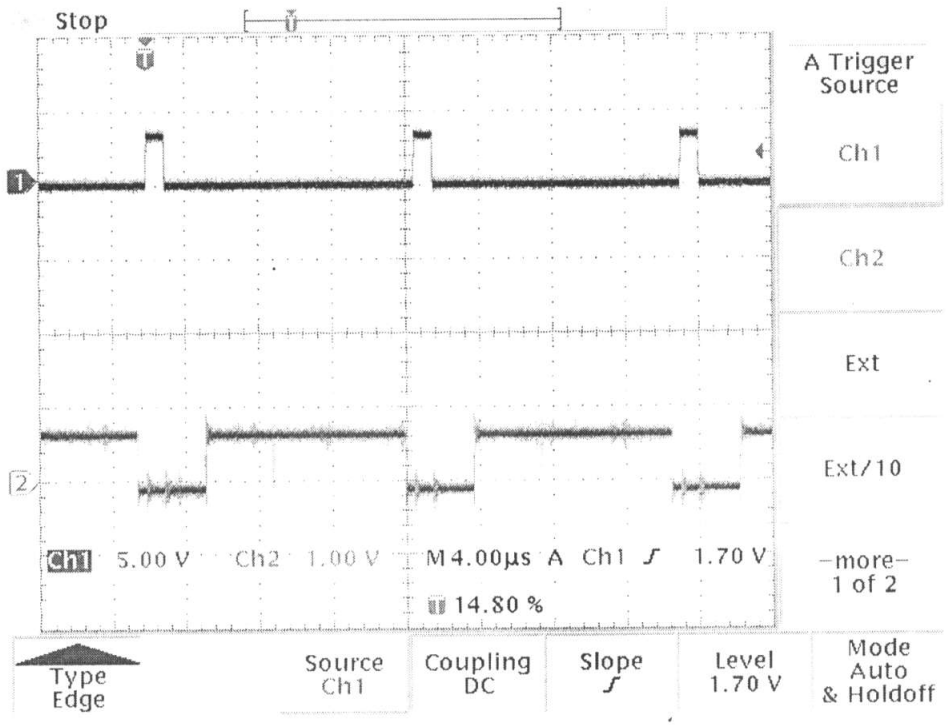


รูปที่ 4.5 รูปสัญญาณ SPI

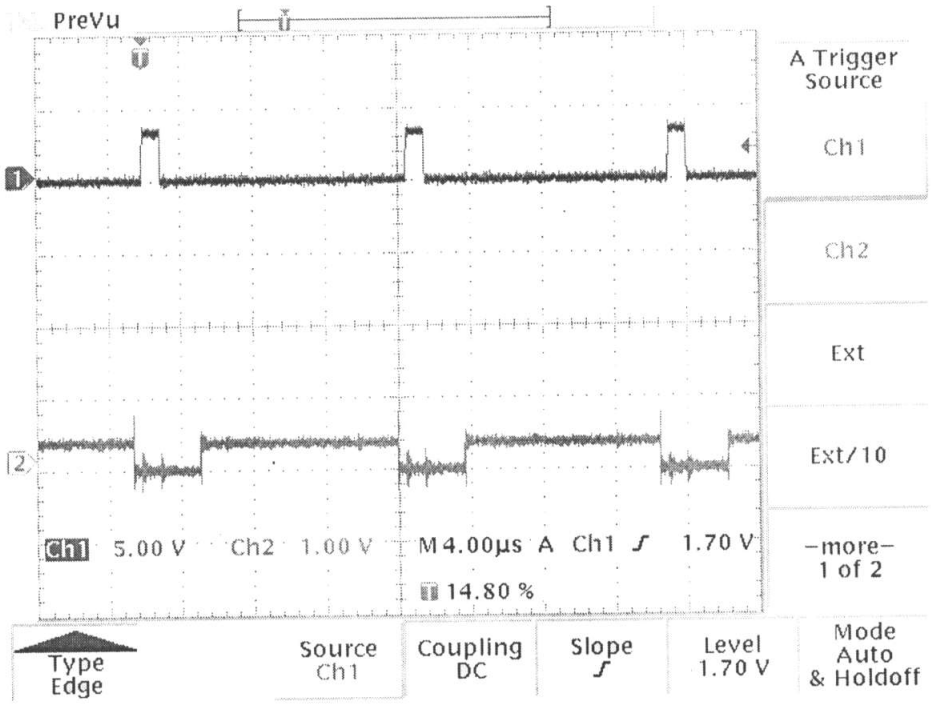


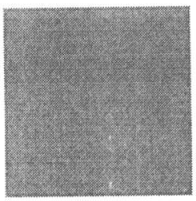
รูปที่ 4.6 รูปสัญญาณ SPI (ส่วนขยาย)

การทดลองที่ 2 วัดสัญญาณของหน้าจอมอนิเตอร์

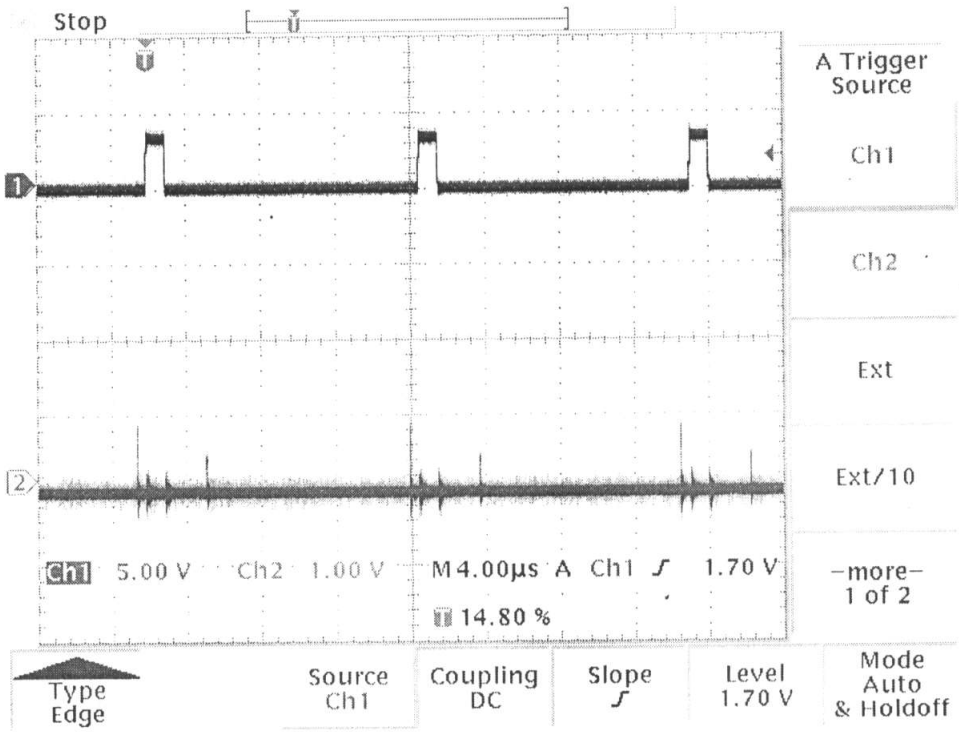


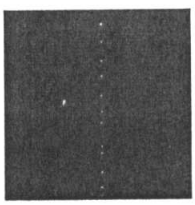
รูปที่ 4.7 รูปสัญญาณสีแดง สีน้ำเงิน สีเขียว (สีขาว)



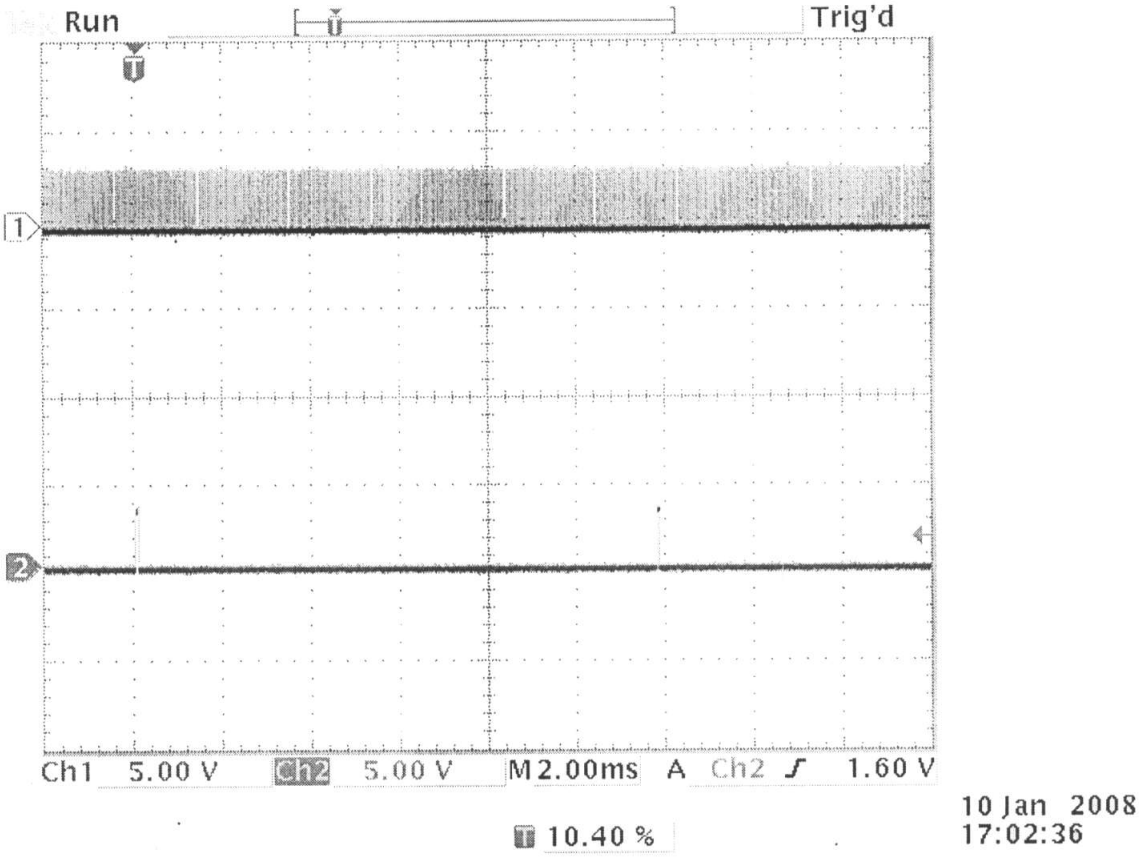


รูปที่ 4.8 รูปสัญญาณของสีผสมที่อ่อนลงมา

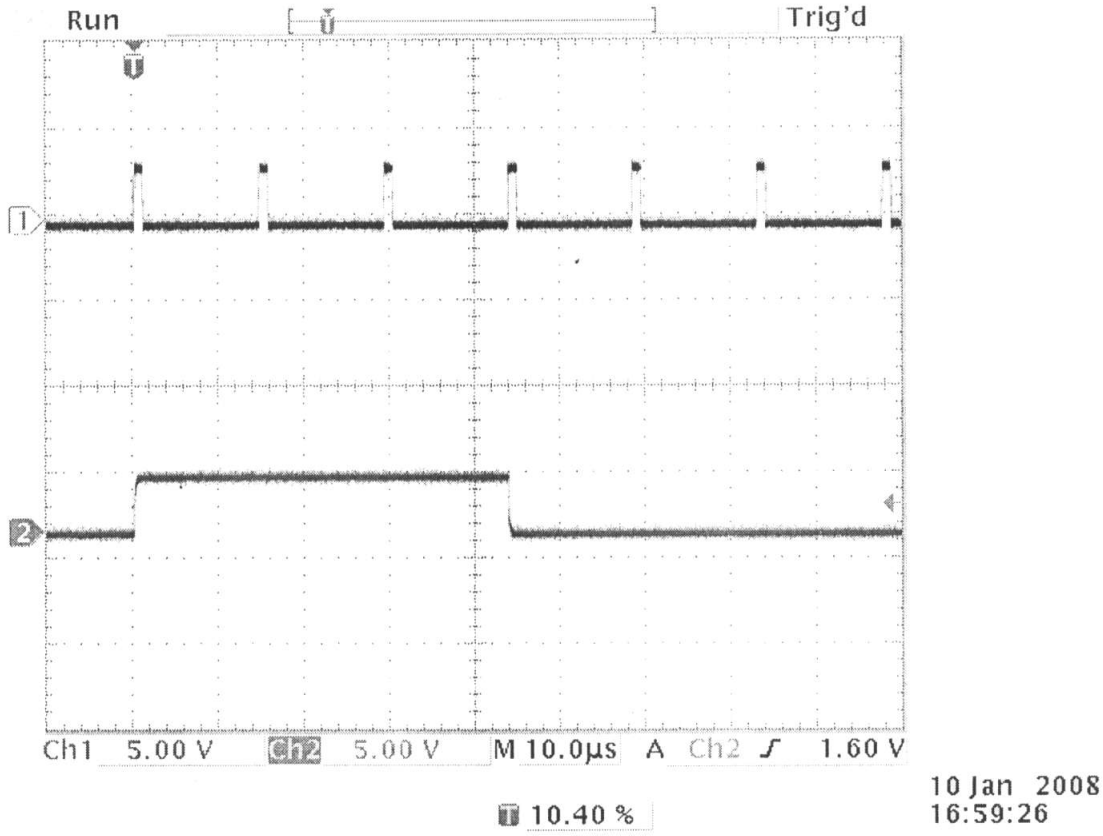




รูปที่ 4.9 รูปสัญญาณของสีดำ

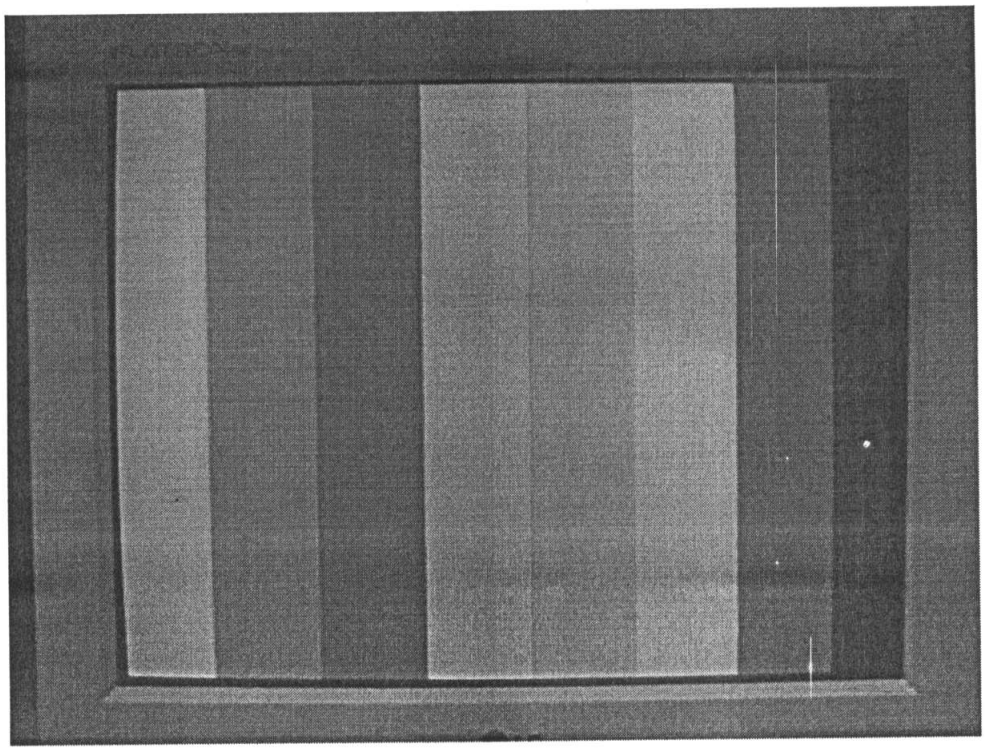


รูปที่ 4.10 รูปสัญญาณ Horizontal sync, Vertical sync

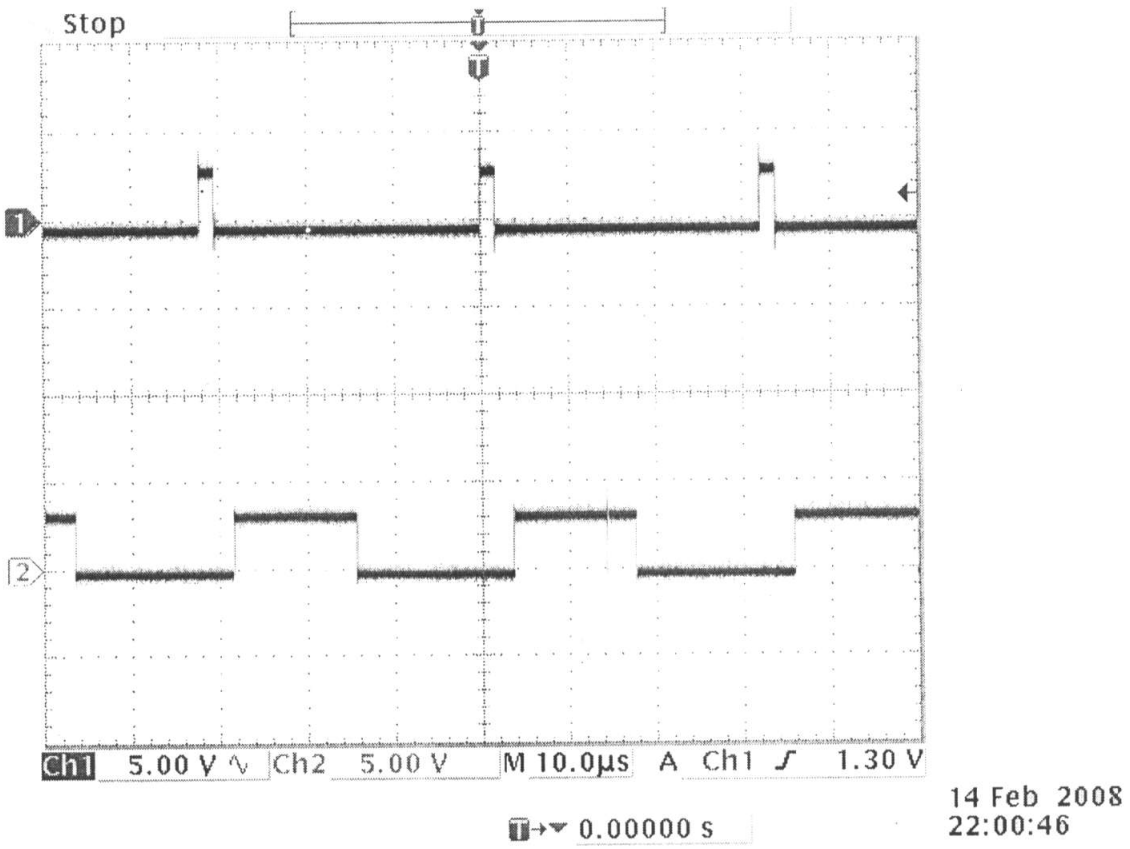


รูปที่ 4.11 รูปสัญญาณ Horizontal sync, Vertical sync (ส่วนขยาย)

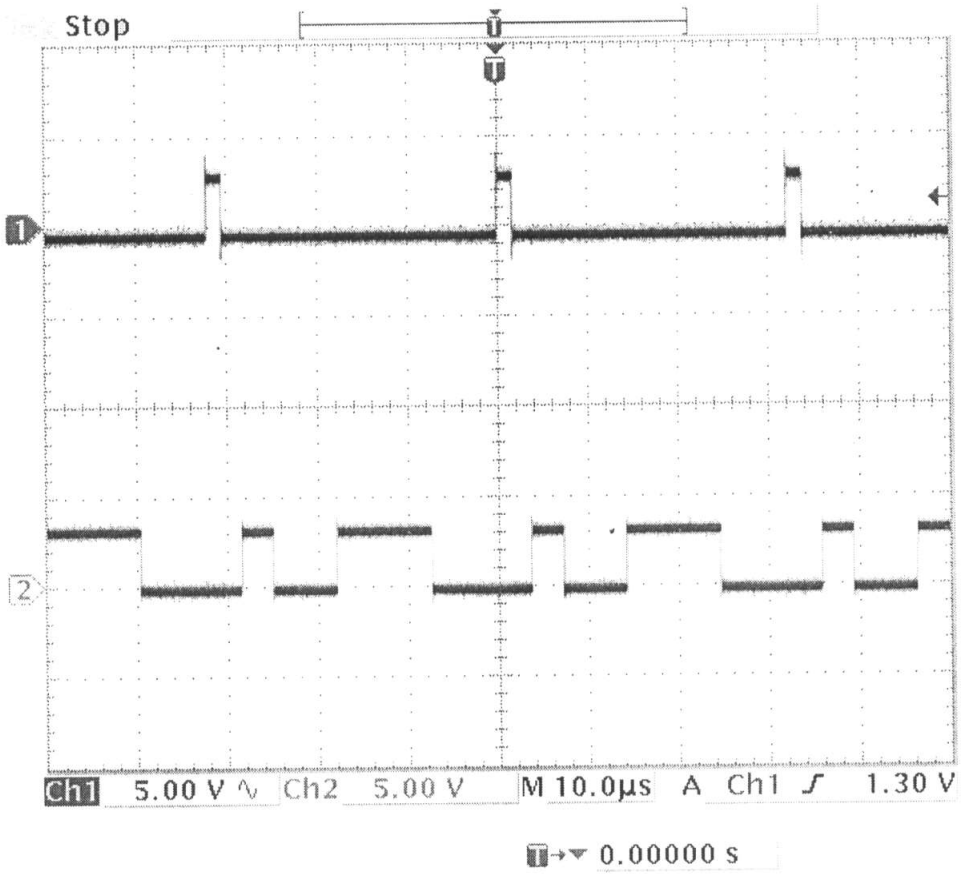
ตัวอย่างภาพ



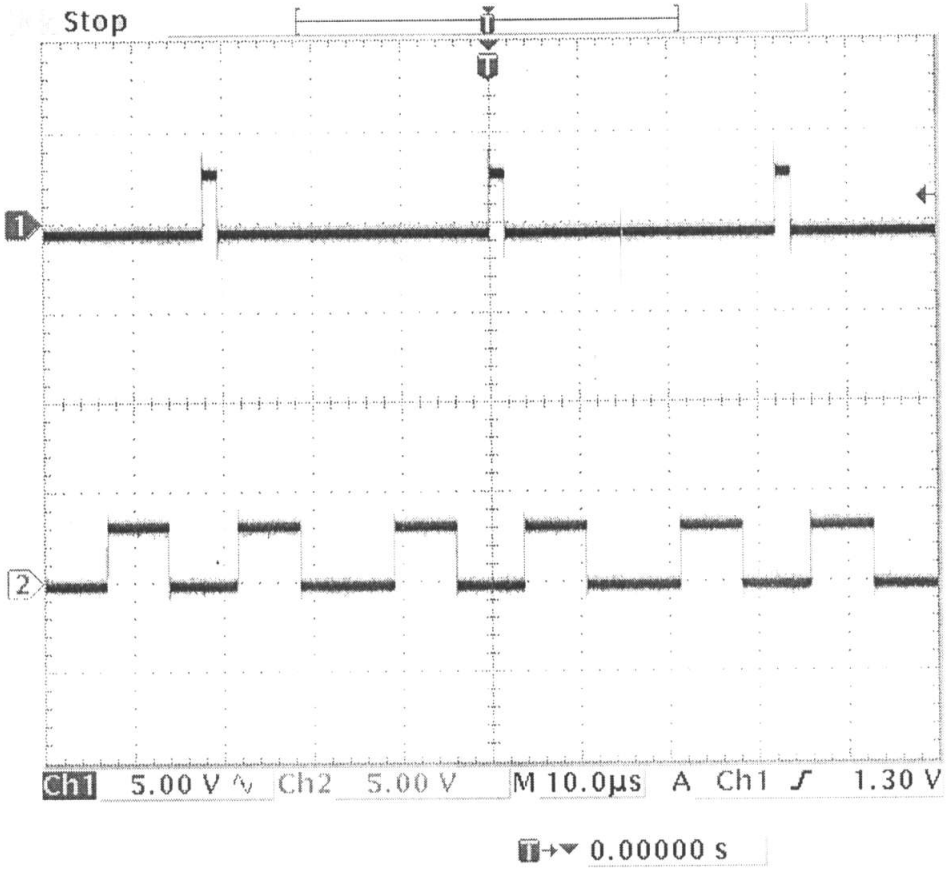
รูปที่4.12รูปแบบแถบสี



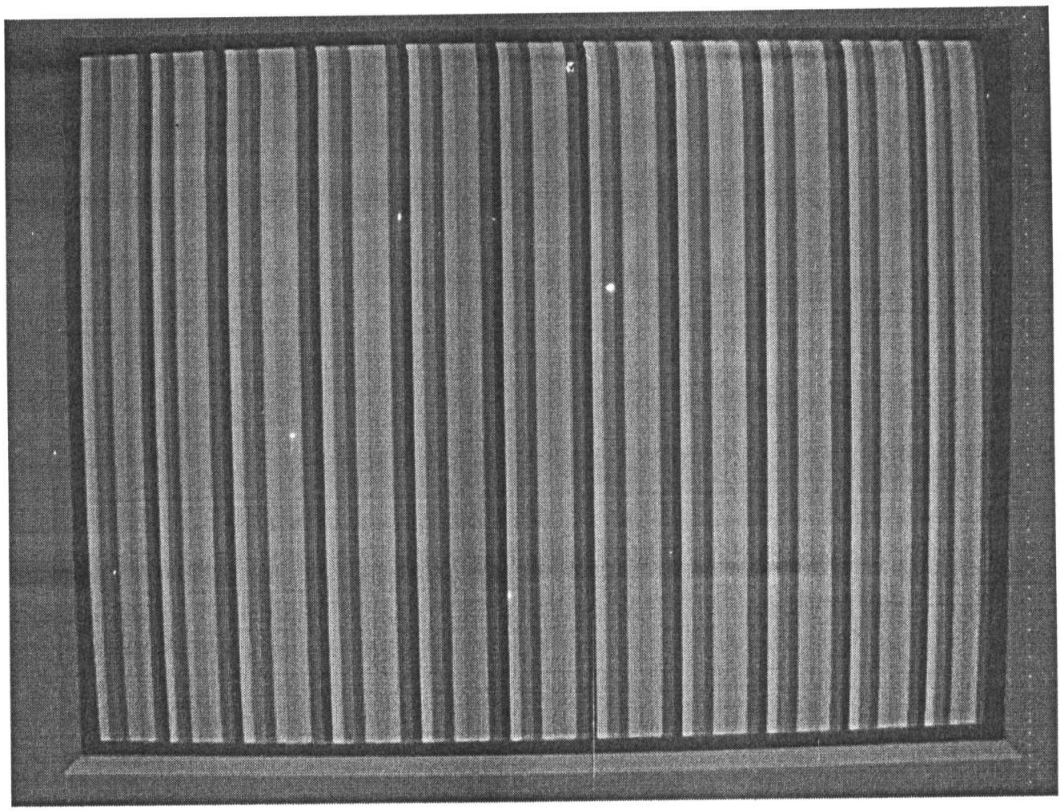
รูปที่ 4.13 รูปสัญญาณ HSync เทียบกับ Red



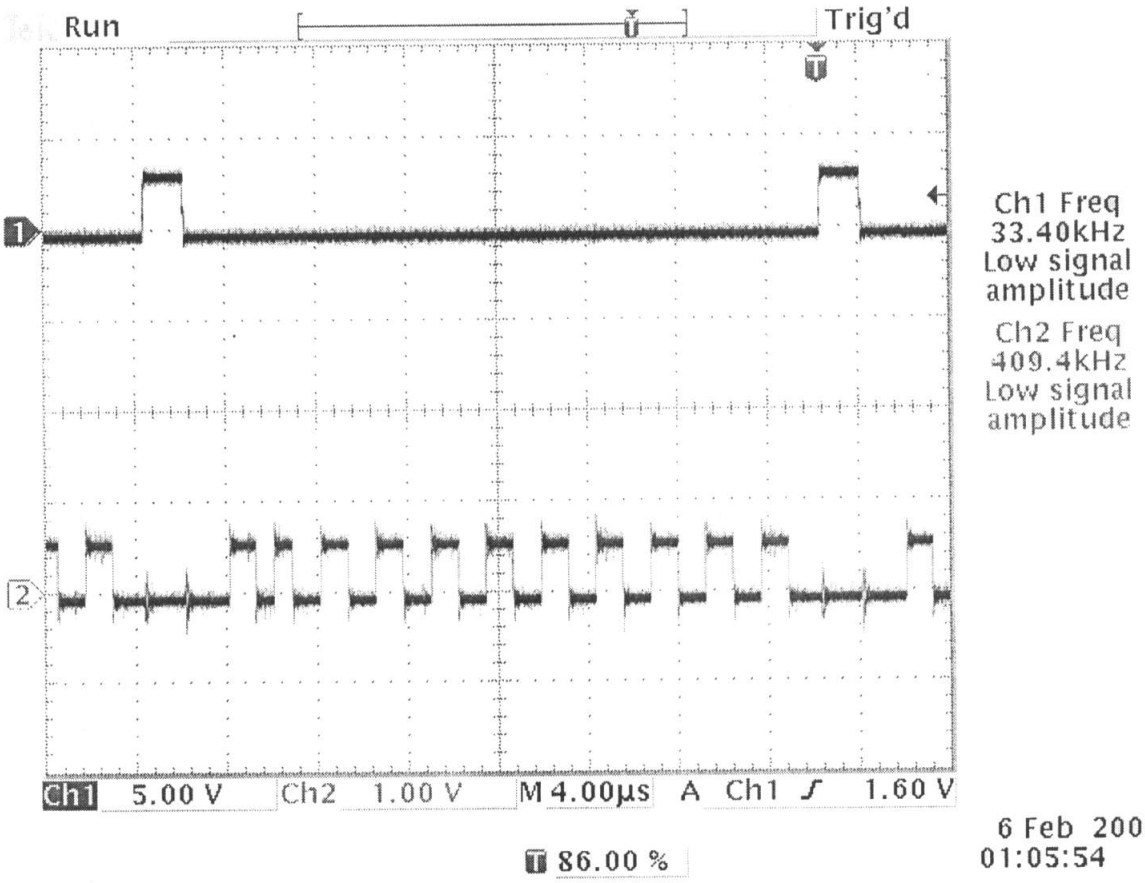
. รูปที่ 4.14 รูปสัญญาณ HSync เทียบกับGreen



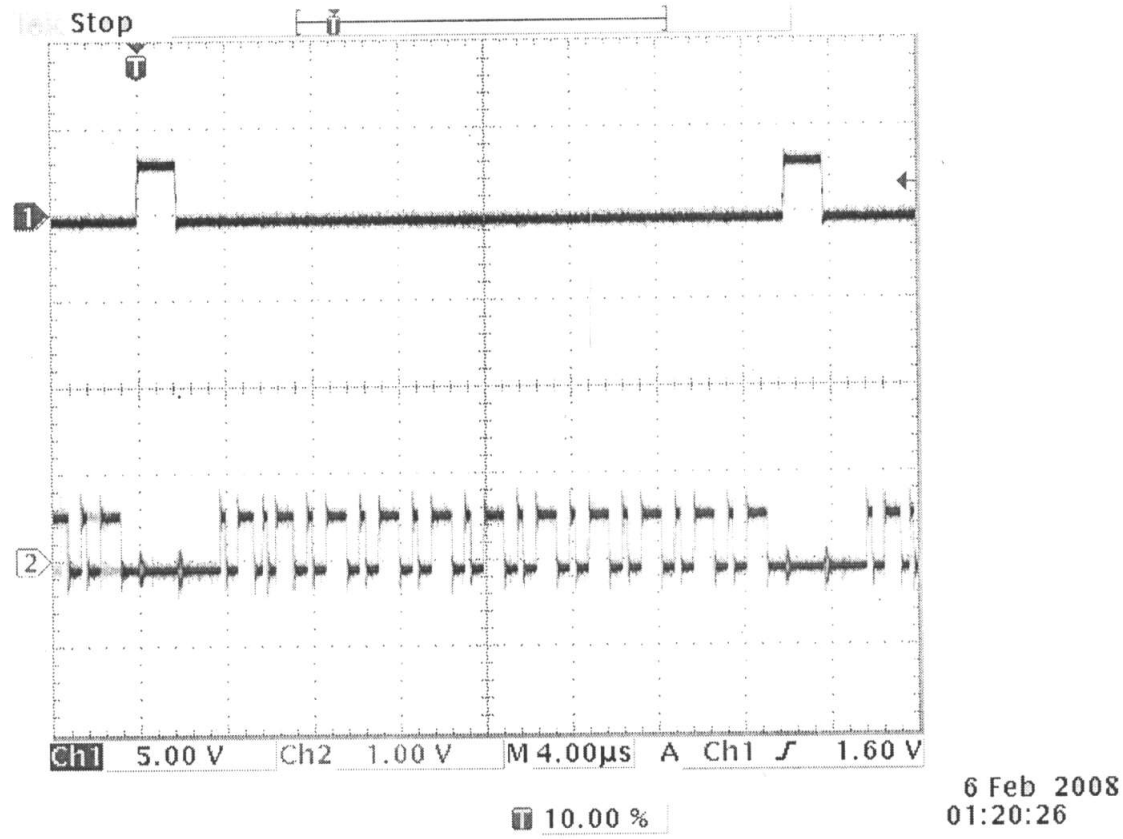
รูปที่ 4.15 รูปสัญญาณ HSync เทียบกับ Blue



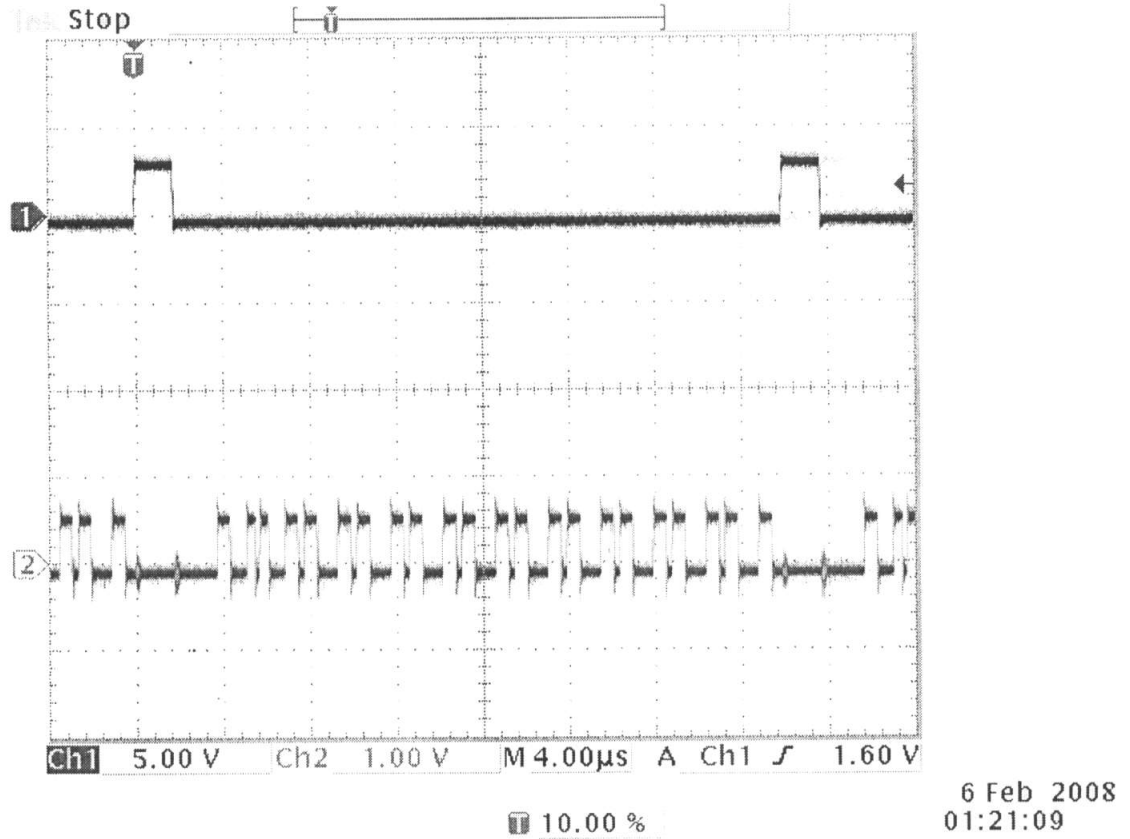
รูปที่ 4.16 รูปแบบแถบสี



รูปที่ 4.17 รูปสัญญาณ HSync เทียบกับ Red



รูปที่ 4.18 รูปสัญญาณ HSync เทียบกับGreen



รูปที่ 4.19 รูปสัญญาณ HSync เทียบกับ Blue

บทที่ 5
สรุป

5.1 สรุป

ในรายงานได้กล่าวถึงความเป็นมาของโครงงาน แนวคิด ทฤษฎี และรายละเอียดการสร้าง อุปกรณ์สร้างสัญญาณภาพสำหรับจอมอนิเตอร์ รวมถึงการทดสอบเบื้องต้น ในการทดสอบการติดต่อระหว่างไมโครคอนโทรเลอร์ (LPC2119) กับตัว SD Card สามารถต่อเข้าโดยตรงผ่าน Port SPI ได้เลย

ผลการทดลองแสดงให้เห็นว่า SD Card สามารถเก็บข้อมูลไฟล์ภาพได้ และแสดงภาพทางหน้าจอมนิเตอร์และสามารถเลือกคุภาพได้

สามารถอ่านไฟล์ภาพได้เฉพาะไฟล์ Bitmap

5.2 ปัญหาและแนวทางแก้ไข

- เนื่องจากอุปกรณ์หลัก (ไมโครคอนโทรลเลอร์ ARM7 และ Socket SD Card) เป็นเทคโนโลยี Surface Mount ทำให้ยากต่อการทดสอบ ผู้จัดทำจึงได้ใช้บอร์ดทดลองของไมโครคอนโทรลเลอร์ตัวนี้ โดยได้ใช้เบอร์ LPC2119

- เนื่องจาก SD Card มีการจัดเรียงข้อมูลในระบบ FAT32 ในการตรวจสอบความถูกต้องจากการอ่านข้อมูลภายใน SD Card จึงทำได้ยาก ดังนั้นผู้จัดทำจึงใช้ซอร์ฟแวร์ Winhex มาช่วยตรวจสอบข้อมูลและตำแหน่ง Address ภายใน SD Card

-เนื่องจาก ARM มีความเร็วในการทำงานช้า ทำให้ไม่สามารถแสดง pixel ของรูปภาพให้มากกว่าการทดลองได้ ดังนั้นจึงต้องเพิ่มสัญญาณนาฬิกาเข้ามาช่วยให้ความเร็วเพิ่มมากขึ้น

5.3 ประโยชน์ที่ได้รับ

- มีความรู้ความเข้าใจในการเข้ารหัสไฟล์ Bitmap
- มีความรู้ความเข้าใจในการแปลงไฟล์ภาพ Bitmap ภายในตัวไมโครคอนโทรลเลอร์
- เนื่องจาก SD Card มีการจัดเรียงไฟล์แบบ FAT32 ทำให้ผู้จัดทำมีความเข้าใจในตัวระบบ FAT32 เพื่อใช้ในการติดต่อระหว่างไมโครคอนโทรลเลอร์กับ SD Card
- มีความรู้ความเข้าใจในการติดต่อสื่อสารข้อมูลในแบบ SPI Bus
- มีความรู้ความเข้าใจในการสร้างสัญญาณภาพแบบอนาลอกผ่านทาง Port D-Sub
- มีความรู้ความเข้าใจในการใช้งานไมโครคอนโทรลเลอร์ตระกูล ARM7