이중 입출력 메모리를 이용한 새로운 영상입력 장치의 설계 및 제작에 관한 연구

吳英洙 · 徐一弘 · 卞增男

차 례

1. 서 론

자동검사, 로봇트의 시각 장치 및 자동 조립 장치 등에 응용되고 있는 실시간 인공시각 시스템을 설계하기 위하여는 주사(Scan) 방식의 TV Camera로 부터 발생하는 아나로그(Analog) 영상 신호를 A/D변환하여 변환된 디지탈(Digital) 영상 정보를 메모리에 입력시키는 장치가 반드시 필요하다. 이와 같은 영상 입력 장치의 설계에 있어서 실시간 조건을 부합시키기 위하여 크게 고려되어야 할 사항으로는 첫째, 아나로그 영상 신호를 Gray 혹은 Binary Level의 디지탈 신호로 부호화 하는데에 걸리는 시간과 둘째, 영상 입력 장치에서 얻은 영상 정보를 중앙 처리 장치(CPU)에 보내는 전송시간이다. 이중에서도 영상 입력 정보의 전송시간 단축이 실시간 처리를 위해 더욱 중요한 바, 그 이유는 고차원(High Level)의 영상 정보 처리를 CPU가 담당하기 때문이다.

지금까지 발표되어온 영상 정보 전송 방법을 보면, (i) 직렬 전송 방법, (ii) 병렬 전송 방법, (iii) DMA(Direct Memory Access) 방법, 등이 있으나 이들은 실시간 정보 처리라는 관점에서 문제가 되고 있다. 구체적으로 살펴보면 한 화면을 256×256의 배열(Array)로 구분하고 한 화소(Pixel : Picture Element)를 256개의 Gray Level(8 bit)로 표현하고자 할 때, 영상 정보량은 64 Kbyte가 된다.

따라서 64 Kbyte의 영상정보를 9600 Baud Rate로 CPU에 보낼때 약 64초가 소요되어, 초당 10회 정도의 판단을 요하는 실시간 처리에 부적합하다. 또한 병렬 전송의 경우도 한 Byte당 50 (μsec) 걸린다고 가정하면 약 3.2(sec)가 걸리게 된다. 한편 DMA 방식은 CPU의 활동시간을 빼앗는 방식이기 때문에 영상 정보 송신 중에 CPU가 정지 상태에 있게 되어 결국 전체적인 성능을 저하시키는 문제가 있을뿐 아니라 DMA 방식을 구현하기 위한 Hardware가 복잡해진다.

이상의 문제점을 해결하기 위하여서는 영상 입력 메모리가 시스템의 주기억 장치(Main Memory)의 역할을 하도록 하면 CPU가 항상 사용 가능케 할수 있게 되어 정보 전송시간 문제가 해결될 뿐만 아니라, CPU에 의한 실시간 Processing도 가능하다.

이와 같은 기능을 갖는 영상 입력 장치를 설계하기 위하여는 두개의 통로(Dual Part)를 갖는 메모리 소자가 필요하다. 그 이유는 한 통로는 System Bus(그림 1)와 직접 연결이 되어야 하고 다른 하나는 Analog 영상 신호를 Digital화 하는 A/D 변환기 및 Digital 영상 정보를 화면에 나타내는데 필요한 D/A 변환기와 연결 되어야만 CPU가 영상 입력 중에도 메모리를 사용할 수가 있기 때문이다.

본 논문에서는 Dual-Port RAM인 TMS 4161를 사용하여 앞서 서술한 문제점들을 해결함과 동시에 주 기억장치의 역할을 할 수 있는 64K-byte의 영상 정보를 입력 저장할 수 있는 새로운 영상 입력 장치의 설계 방안을 제시 하고자 한다.

특히 TMS4161 메모리의 제어 회로중 지금까지 그 설계 방법이 발표되어 있지 않은 〔3〕버스 중재기를 8203 DRAM Refresh Controller 를 이용하여 설계하는 방법을 제시하고자 한다.

또한 제시된 버스 중재기는 물론 A/D, D/A

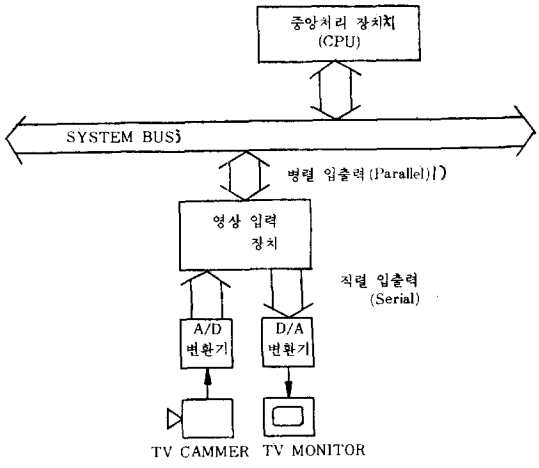


그림 1. 영상 입력 장치의 기본 구조

변환기 및 동기 신호 발생기 등을 설계제작하고 제작된 영상 입력 장치를 MC68000 CPU에 연결하여 실험함으로써 제시된 방법의 타당성을 입증하고자 한다.

2. 영상신호 표본 추출시간과 영상 메모리의 요구조건

영상입력 장치의 센서(Sensor)인 TV Camera가 만들어내는 화면(Video Frame)의 구성 성분을 보면 그림 2와 같다.

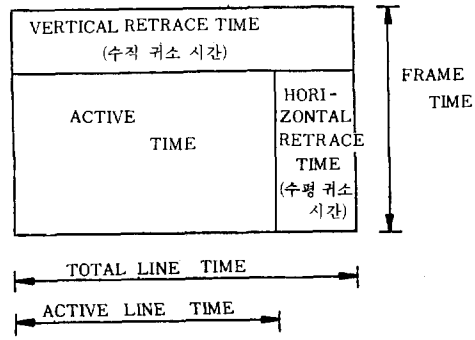


그림 2. 화면의 시간적 구성 성분

여기서 보통 TV 방송에서 사용되고 있는 RS 170A의 경우를 살펴보면 Frame Time이 1/60초 (Non-interlace 때), Total Line Time이 63.556 (μsec), 수직귀소 시간이 20x(Total Line Time) 수평 귀소시간이 10.9(μsec)이다. 한편 영상신호를 Digital와 할때 필요한 정보로는 한 Frame에서의 Scanline 갯수, 한 Line에서의 Pixel 수, 그리고 한 Pixel을 나타내는 bit수이며, 이들을 각각 l, p 및 d라고 할 때 Ixp는 해상도(Resolution)를 나타낸다.

이제 상기한 영상 신호의 구성 성분 및 해상도를 이용하여 한 Pixel을 Digital화 하는데 걸리는 시간인 Pixel Time을 유도하면 다음과 같다.

$$\text{PIXEL TIME} = \frac{\dfrac{1}{\text{REFRESH RATE}} - \text{VERTICAL RETRACE TIME}}{\text{ACTIVE LINES PER FRAME}} \Big/ \text{ACTIVE PIXEL PER LINE} \quad (1)$$

— VERTICAL RETRACE TIME

예를 들어, 현재 가장 일반적으로 산업용 비

전 시스템의 채택되고 있는 해상도인 256×256으로 Digital화 할 경우 Pixel Time은 식(1)로부터.

$$\frac{\text{PIXEL}}{\text{TIME}} = \frac{63.556(\mu\text{sec}) - 10.9(\mu\text{sec})}{256}$$

$$= 206\,(\text{n sec})$$ 가 된다.

보통의 Memery Device인 Dynamic RAM이나 Static RAM으로는 Pixel Time으로 정보를 저장시킬 수가 없다. 왜냐하면 상기한 메몌리의 Cycle Time이 206(nsec) 보다 큰 250~400(nsec)이기 때문에 영상 정보를 206(nsec)의 Pixel Time으로 Memory에 저장시킬 경우 영상 정보가 손실되기 때문이다.

그러므로, Video 정보를 손실없이 기억 장치에 넣기 위한 새로운 메모리가 필요하다. 이와 같은 요구 조건을 만족시키기 위하여 개발된 것이 미국 TI사의 TMS4161로서 이는 속도가 빠른 Shift Regiater를 이용하여 영상정보를 Buffering 시킨 후 한꺼번에 DRAM에 저장시키는 방법을 사용한다. 특히 TMS4161은 256bit의 Shift Register를 직렬로 연결하여 사용하고 있는 바, 그 이유를 살펴보기로 한다.

먼저 그림 3과 같이 4개의 Shift Register를 사용한 경우를 살펴보면, Shift Register는 매 200(nsec)마다 한번씩 Shifting을 행하여 4 Pixel을 저장하므로 저장시간은 800(nsec)가 된다. 따라서 800(nsec)가 경과한 후에 DRAM과 같은

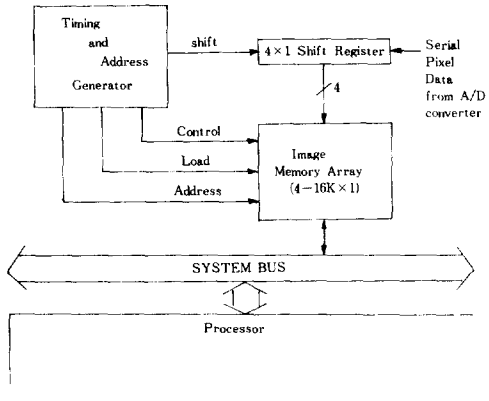


그림 3. Memory소자의 Cycle Time을 만족시키기 위한 방법

Memory로 Shift Register의 내용이 전달(Loading)되기 때문에 400(nsec)의 Cycle Time을 갖는 Memory로서, Data의 손실없이 네 Pixel Time마다 Data를 저장할 수가 있다.

만일 8 bit Shift Register를 사용한다면 메모리가 더욱더 많은 시간적 여유를 갖게 되므로 남는 시간에 CPU가 Video Data를 읽어서 처리할 수가 있다. 그러므로 Shift Register의 bit 수를 많이 하면 할수록 CPU가 Image Memory를 사용할수 있는 시간이 많이 생기게 되고, 특히 Shift Register의 크기가 한 Scan Line의 Pixel갯수와 일치하게 하면, 직렬 Pixel Data가 Shift Register에 저장되는 시간동안 CPU가 Image Memory를 주기억 장치와 같이 사용이 가능하게 된다.

이러한 특성을 갖는 메모리가 TMS4161이며, 따라서 본 논문에서는 CPU의 주기억 장치의 역할을 할 수 있는 256×256의 해상도를 갖는 영상 입력 장치를 설계 제작키 위하여 TMS 4161을 사용하기로 한다.

3. 이중 입출력 메모리 (Dual-port RAM) TMS4161의 구조 및 제어 신호기 설계 문제

본절에서는 제2절에서 간략히 기술한 TSM 4161의 구조 및 입출력 신호의 Timing Dingram을 살펴봄으로써, TMS4161의 제어를 위한 버스 중재기의 설계문제를 고찰하고자 한다.TMS4161의 내부구조는 그림 4에서 보는 바와 같이 크게 두 부분으로 이루어져 있다. 그 중 하나는

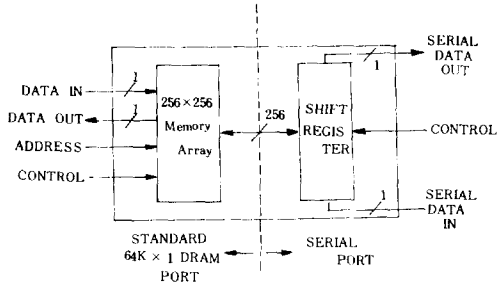


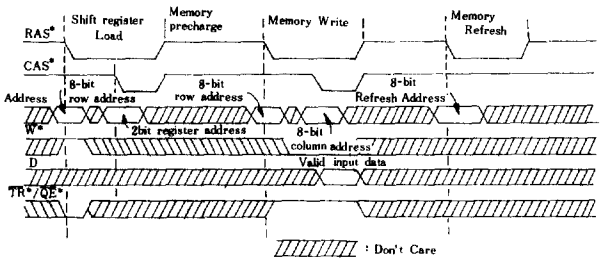
그림 4. Dual-port RAM(TMS4161)의 구조

기존의 64K×1 DRAM으로서 CPU로 부터 Data를 입출력 할 수가 있으며, 다른 하나는 256 bit의 Shift Register로서 Serial Data를 입출력 할 수 있다. 또한 64K DRAM Port와 Serial Port 동시에 Accessing이 가능하다.

64K×1 DRAM부분은 256×256×1의 Memory Array로 구성되어 있으며 256개의 Row가 각각 Shift Register와 제어 신호에 의하여 연결이 가능하게 되어 있다. 한편, 영상 Data를 이 Memory Array에 넣기 위해서는, 먼저 Shift Register에 Data를 넣은 후 Row Address로 지정된 곳에 전달(Transferring)만 하면 된다. 반대 과정을 수행하면 Memory Array에 있는 내용을 Shift Register로 옮기 후 Serial Output 단자로 내보낼 수가 있다.

좀더 구체적으로 보면 TMS4161에는 그림5의 Timing Diagram에서 보는바와 같이 5가지 동작 Cycle이 있다.

A. Random Read Cycle
B. Random Write Cycle
C. Shift Register to Memory Transfer Cycle
D. Memory to Shift Register Transfer Cycle
E. Refresh Cycle



Note; $\overline{(\cdot)}$, $(\cdot)^*$는 Active Low 신호를 나타낸다.

그림 5. TMS4161의 Timing Diagram

따라서 이상 5가지의 TMS4161의 동작을 원하는 시간에 적절히 수행이 되도록 제어하기 위한 신호를 발생 시켜야만한다. 이를 위하여 TMS4161의 제어 신호 단자의 특성을 살펴보기로 한다.

(TR/QE)*단자는 TMS4161의 5가지 동작중 메모리 관련 동작 (A 및 B)과 Shift Register관련동작(C 및 D)을 구별하는데 사용하며, (W)* 단자 신호는 메모리 관련 동작의 경우는 A와 B를 구별하는데, Shift. Register관련동작에 대해서는 C와 D를 구별하는데 사용한다. 구체적으로, (RAS)*(Row Address Strobe)단자 신호의 Falling Edge에서 (TR/QE)* 단자 신호가 "L"이면 Shift Register와 관계되는 동작을 수행하며, 이 때의 Address A0~A7이 Memory Array의 Row Address를 나타낸다. 또한 이때의 (W)*의 상태가 Data의 전달 (Transfer)방향을 나타내게 되고 (W)*="L"이면 SR (Shift Register)에서 A0부터 A7에 의하여 지정된 메모리의 한 Row로, (W)*="H"이면 A0~A7에 의하여 지정된 Memory Array의 한 Row에서 SR로 전송이 된다. 한편 (CAS)*(Coulomn Address Strode)의 Falling Edge에서의 A7, A6은 전송될 Bit수를 결정하는 것으로서 A7A6=00이면 256 bit, 01이면 192 bit, 10이면 128 bit, 11이면 64 bit가 전송이 된다.

(SOE)*(Serie Output Enable)단자는 여러개의 Shift Register가 함께 연결되어 있을 때 그것들의 출력을 제어하는데 사용되며, A 및 B의 경우인 메모리 관련 동작 제어는 기존의 DRAM 제어와 동일하게 할 수 있다.

한편 TMS4161의 내부구조가 Dynamic RAM의 형태로 되어 있으므로 Refresh는 반드시 해주어야 하는 바, 이 때의 Refresh방법은 기존의 DRAM과 같이 (RAS)*만에 의하여 ((RAS)*Only Refresh)행할 수 있다.

이제 상기 기술한 제어단자의 특성을 이용하여 TMS4161로 하여금 원하는 동작을 수행하도록하는 제어 신호 발생기의 설계 문제를 기술하고자 한다.

TMS4161의 메모리 주소를 지정하는 단자들

이 한 Set(8 bit)인 반면에, (1) CPU가 지정하고자 하는 메모리 주소, (2) Shift Register가 지정하고자 하는 메모리 주소 및 (3) Dynamic RAM의 경우 반드시 필요한 Refresh용 메모리 주소가 각각 다르며, 요구시간 및 요구되어지는 제어신호의 Timing Diagram이 그림 5에서 보는 바와 같이 다르다.

그러므로 설계 문제는 첫째 CPU, Shift Register 및 Refresh때 필요한 주소신호 발생기를 설계하는 것과, 둘째 CPU, Shift Register 및 Refresh가 TMS4161의 Address의 사용을 요구할 때, 충돌이 일어나지 않도록하는 버스 중재기 설계 및 각 동작에 필요한 제어신호 발생 장치를 설계하는 것이다.

4. Dual-port RAM 제어기의 설계

제 3 절에서 기술한 주소버스중재기 및 제어신호 발생기의 설계 문제를 해결하기 위하여 먼저 다음과 같이 용어를 정의하기로 한다.

CPU의 Address 신호를 MEMADDR (Memory Address), Shift Register를 SR, SR과 관련된 Memory Array의 Row Address를 LINADDR (Line Address)라 하고 위의 세가지의 Address 사용 요청을 나타내는 신호를 각각 MEMREQ (Memory Request), LINREQ (Line Request) 및 REFREQ (Refresh Request)라 하자. 또한 각 Address사용 요청 신호에 해당되는 Cycle 구간을 각각 MEMCYC (Memory Cycle), LINCYC (Line Cycle) 및 REFCYC (Refresh Cycle)이라 하고 Shift Register의 크기를 나타내는 두 bit의 Address A7 및 A6를 SFTADDR (Shift Address)라고 정의하기로 한다.

이제 정의된 용어를 사용하여 설계문제를 기술하면, 그림 6과 같이 MEMADDR, LINADDR 및 REFADDR의 세가지의 Address Line 및 Request 신호인 MEMREQ, LINREQ, REFREQ 등을 입력으로 하여, 출력에 세가지 Address중 먼저 요청된 Address와 그림5의 Timing Diagram을 만족시키는 TMS4161 제어 신호가 되도록 하는 Address Multiplexer와 요청중재기 (Request Arbitor)로 구성된 제어 장치를 설계하는 것이다.

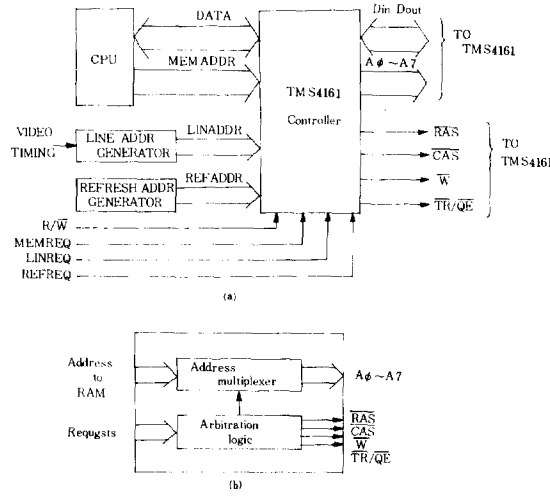


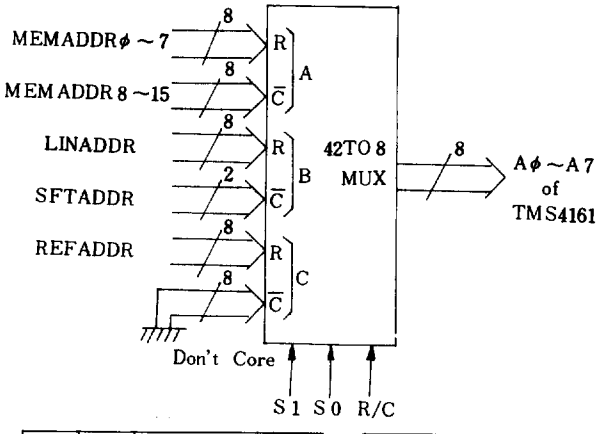
그림 6. (a) TMS4161 (Dual-Port RAM) 제어기의 역할
(b) 내부적으로는 Address Mux와 중재기 (Arbiter)로 이루어진다.

4.1 주소 신호 발생기의 설계

3가지 주소 신호의 Muitiplexer를 그림7에서 보는 바와 같이 설계하고자할 경우, 설계개념 은 간단하나 Multiplexer의 크기가 방대해질 뿐만 아니라 주소사용 요구신호 발생시에, 어떤 주소를 선택할까를 결정하는 S1, S0 및 R/C의 신호 발생방법이 쉽지않는 문제가 된다. 더우기, Dynamic RAM Refresh 동작은 주기적으로 행하여 주어야 하는 반면에 실제의 Data의 흐름에는 관여하지 않는 부분이므로 한꺼번에 3가지를 구별하고자하는 상기 설계방법은 비효율적이다.

따라서 본 논문에서는 TMS4161도 기존의 표준 64K×1 DRAM과 같은 내부구조를 가지고 있다는 점을 고려하여 기존의 DRAM 제어용 VLSI인 Intel8203를 사용하고자하며, 이를 이용하여 Refresh문제를 해결함과 동시에 Multiplexing을 두개의 주소씩 계층적으로 할 수 있도록하여, 상기한 방법에서의 문제점을 해결하고자 한다.

이를 위하여 먼저 그림 8에서와 같이 8203의 내부 Block Diagram 및 Refresh Cycle과 Read /



S1	S0	
0	0	MEMADDR (A)
0	1	LINADDR (B)
1	0	REFADDR (c)
1	1	Don't Care

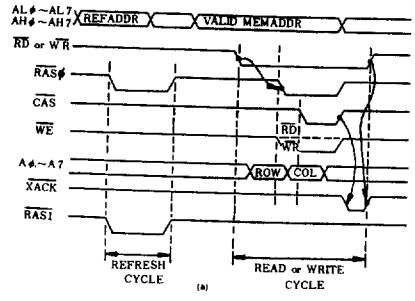
R/C	
1	ROW ADDR
0	COLUN ADDR

그림 7. Address Multiplexer

Write Cycle의 Timing Diagram을 나타내었다.

8203은 내부적으로 일정한 시간마다 Refresh 제어신호를 발생하여 DRAM Refresh를 자동적으로 행하여준다. 또한 DRAM의 Read 혹은 Write요청과 내부적 Refresh 요청을 중재하여 내부적 Refresh Cycle의 수행시에는 RASφ* 및 RAS1* 신호를 동시에 "L"로 보내내고, Rendom Access Cycle시에는 (XACK)* 신호를 내보내 Cycle의 종료를 CPU에 알려주게 된다. (그림 8 (a)) 이제 그림 8 의 8203 Timing Digram과 그림 5 의 TMS4161의 Timing Diagram을 비교하여보면 Refresh때와 Memory Read 및 Write Cycle은 동일함을 알 수 있다. 따라서 TMS4161의 3 가지 주소사용요구 신호에 대하여 각각 발생시켜 주어야만 하는 MEMCYC, REFCYL 및 LINCYC중에서 MEMCYC 및 REFCYC를 8203을 이용하여 발생 시킬수가 있다.

이를 구체적으로 하기 위하여 먼저 그림 8 로부터 8203이 내부적으로 REFCYC과 MEMCYC을 중재 (Arbitration) 하여 수행하는 바 내부적인 REFCYC중에는 (RAS 0)*와 (RAS 1)*이 동시에 Active 상태가 되고 MEMCYC의 종료시에는



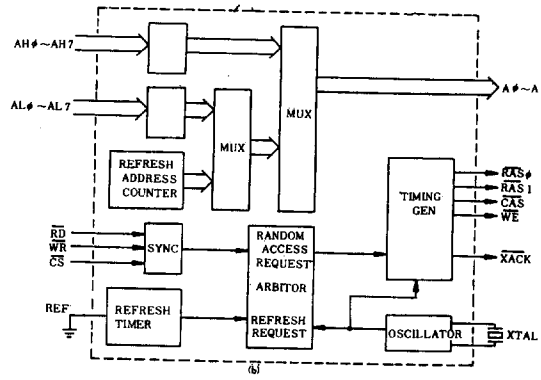


그림 8. (A) 8203의 Refresh Cycle 및 Read/Write Cycle Timing Dingram Refresh Cycle동안에는 (RAS)*와 (RASI)* 이 동시에 "L"로 된다.
(B) 8203의 Block Diagram (64K Mode)

(XACK)* 신호를 Activation시킨다는 것을 주목하므로써, 현재의 8203의 상태가 내부적 (Internal) Refresh Cycle 구간임을 (RAS 0)* 및 (RAS 1)* 이 모두 ACTIVE상태임을 검사하므로써 찾아낼 수 있음을 알수 있다. 따라서 이들 두 신호로 부터 TMS4161의 Refresh용 (RAS)* 를 만들어낼 수 있다. 한편 Memery Cycle (MEMCYC) 의 경우도 CPU의 Read 혹은 Write요청에 의하여 (RD)* 혹은 (WR)* 신호를 기동 (Activ-

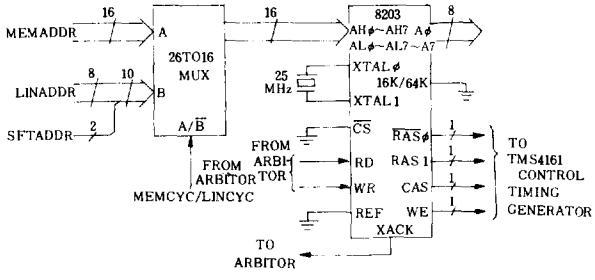


그림 9. DRAM Controller (Intel 82 03)을 이용한 TMS4161 제어기

ate)시킴으로써 시작 시킬수 있고(XACK)* 신호에 의하여 MEMCYC이 끝났음을 알 수가 있으므로 MEMCYC도 8203을 이용함으로써 해결을 할 수가 있는 것이다.

이상과 같이 REFCYC 및 MEMCYC이 8203을 사용하므로써 해결이 될 수 있으나 LINCYC의 경우는 기존의 DRAM에는 없는 제어신호(TR/QE)를 발생시켜야 하고 또한 MEMCYC과의 중재를 행하여 주어야 하기 때문에 8203만으로는 해결할 수가 없다. 그러므로 LINCYC의 해결을 위한 설계방법을 찾아야만 한다. 이를 위하여 본 논문에서는 LINREQ와 MEMREQ에 따른 중재기 및 Multiplexer를 별도로 설계를 해주어, 전체적으로 먼저 LINREQ와 MEMREQ 를 중재하고, 이 둘 중 하나와 REFREQ를 8203이 중재하도록 하고자 한다.

4.2 요구 중재기 (Request Arbiter)의 설계

LINCYC(Line Cycle)와 MEMCYC(Memory Cycle)은 동시에 요청이 될 수가 있으므로 반드시 중재(Arbitration)를 하여 주어야만 하고 중재 결과에 의하여 각 요청에 맞는 Timing을 만들어 주어야 한다. 이를 위하여 각 신호 특성을 살펴보기로 한다.

먼저 Memory Request(MEMREQ)의 경우를 살펴보면, CPU는 Address Bus에 Image Memory의 Address를 내보내게 되고, (R/W)* 신호에 의하여 Read 혹은 Write할 것인가를 나타낸다. Address Map Decoder에서는 Address가Image Memory공간 일때 MEMREQ신호를 만들어 내어서 중재기에 메모리 사용을 요청하므로써 MEMCYC은 시작이 된다. 그리고 LINCYC는 한 Scan Line의 처음 혹은 끝에서 행하여 주면 되므로 LINREQ 신호는 HSYNC(Horizontal Sync) 신호와 CB(Composite Blanking) 신호에 의하여 결정된다. 이상과 같이 발생된 MEMREQ과 LINREQ의 두 신호의 중재를 행하여 어느것을 먼저 수행해 줄것인가를 결정하는 것이 중재기의 역할로서, 그림10의 흐름도에서와 같이 어떠한 Cycle이든지 간에 Cycle도중에 사용 요청이 왔을 경우에는 그 요청을 기다리게 해놓고(Pending) 현재의 Cycle을 끝냄과 동시에 기다리고 있는 요청을 수행하도록 해야한다.

그림 9의 흐름도를 Hardware로 설계한 것이 그림11이다.

그림11에 대하여 설명하면 다음과 같다.

FF 1 및 FF 2는 Request Pending Flip-Flop이고 FF 3는 어떤 요청이 요구되어 왔는지를

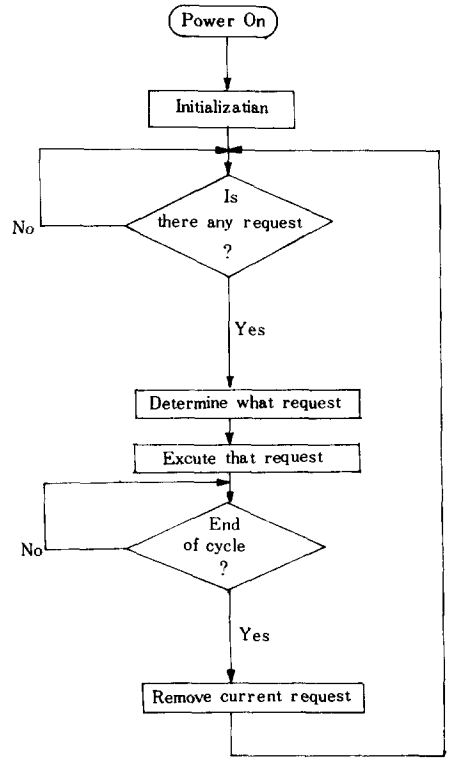


그림10. Request Arbitor Flow Chart

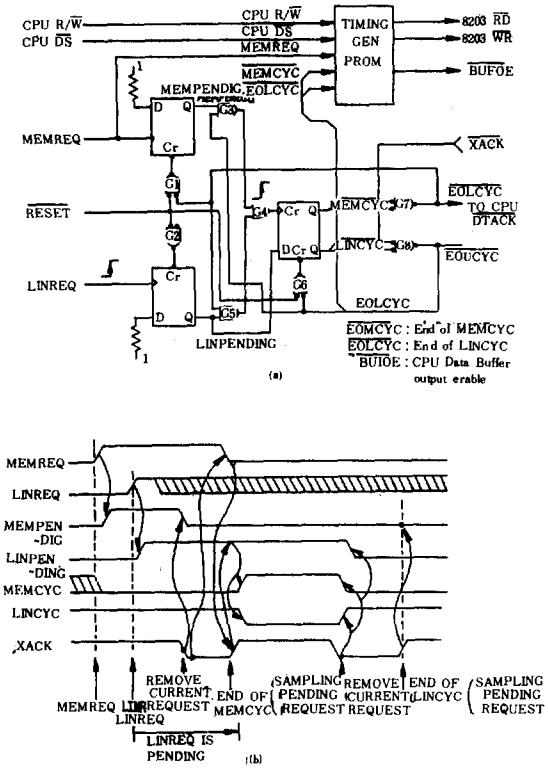


그림11. Request Arbitor 및 Timing Diagram

판별하는 Flip-Flop이다. 먼저 MEMREQ(LINREQ)신호의 Rising Edge에서 FF1(FF2)의 출력Q가 "H"(Logic 1)로 전이(Transition)되면 OR-Gate에 의하여 FF3의 CK(Cleok)단자에 신호가 가해짐과 동시에 FF2의 출력Q를 Sampling 한다.

만일 FF3의 Ck의 발생이 FF2의 출력 Q에 의한 것이면 FF3의 출력(Q)*는 "L"(Liogc)가 되어 LINREQ의 요청이 있음을 판별하게 되고, FF3의 Ck의 발생이 FF1에 의한 것이 있으면 MEMREQ라는 것을 알 수 있다. FF3의 Ck단자는 AND 및 OR Gate를 통과하는 Gate지연(Delay)에 의하여 FF3의 D 입력의 Setup Time은 항상 만족하게 되어서 FF3의 불안정한 상태는 없게된다.

그림8에 나타낸것과 같이 8203은 한 Cycle이 끝남과 동시에 (XACK)*(Transfer Ackneledge) 신호를 Activation시킨다. 이 신호에 의하여 현재의 요청을 수행하였다는 것을 알수 있고 그림10의 G7혹은 G8에 의하여 현재 요청의 끝을 알아냄과 동시에 G6에 의하여 FF3를 지우고 G1 및 G2를 통과시켜 요청된 Pending FF를 지움과 동시에 G3혹은 G5에 다른 요청이 기다리고 있는지를 살펴본다. 만일 다른 요청이 있으면 (XACK)*의 Rising edge에 의하여 FF3가 전이됨과 동시에 그 요청에 대한 수행을 하여주게 된다.

전원 공급시에는 FF1, FF2 및 FF3가 모두 Reset되어 Pending된 요청은 없고 현재의 Cycle상태는 MEMCYC임을 나타낸다. 그러나 FF3의 상태변화는 CK단자의 Edge에서 일어나게 되므로 현재의 Cycle상태가 MEMCYC을 나타내어도 무방하며, 또한 Timing Generator에서 MEMREQ신호에 의하여 조건화(Conditioning)이 되므로 초기화는 정상적으로 수행되는 것이다.

위의 설명한 과정을 MEMREQ가 요청되고 MEMREQ의 서비스를 수행중에 LINREQ가 요구될 경우에 대하여 그림11의 (b)에 Timing Diagram을 나타내었다.

4.3 Dual-Port RAM제어기 및 제어 신호 발생기의 설계

본절에서는 4.1절에서 기술한 주소신호발생기와 4.2절에서 설명한 요청중재기(Requet Arbiter)를 이용하여 TMS4161의 제어기 및 제어신호 발생기의 설계를 하고자 한다. 그림12 에 주소신호 발생기 및 요청 중재기를 이용한 Dual-port RAM제어기의 입출력 신호를 나타내었다. 여기서 제어기의 입력으로는 MEMREQ 신호와 LINREQ신호 및 그에 해당하는 주소 신호이며 출력으로는 (RAS)*, (CAS)*, (TR/QE)*, (W)* 및 A0～A7이다. (그림12)

이와 더불어 MEMCYC일때의 정보의 방향을 나타내는 CPU R/(W)* 신호도 입력으로 작용하여야 하고, LINCYC일때도 A/D 변환된 정보를 Memory Array에 넣을 것인가(Acquis,tion)혹

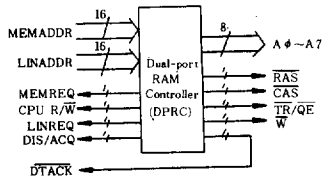


그림12. 구성하고자 하는 Dual-Port RAM (TMS4161) 제어기의 입력 및 출력

은 Memory Arrey의 내용을 Display할 것인가를 결정하는 DIS/(ACQ)*라는 보조적인 입력 신호가 있어야 한다. 이 신호는 CPU Address 공간 중에서 하나의 제어 레지스터(Control Register)를 설정함으로써 만들어 낸다. 또한, 이 제어 Register에는 SFTADDR의 2 bit도 포함되도록 하여 CPU에 의하여 제어가 가능하도록 한다.

한편, MEMCYC일때에 정보의 전달이 종료되었음을 CPU에 알려주는 (DTACK)*(Data Trar Transfer Acknowledge)신호가 출력으로 나오게 하여야 하는 바, 이는 그림11(a)에서 보는 바와 같이 8203의 출력 (XTCK)* 신호와 (MEMCYC)*신호의 AND 작용에 의하여 쉽게 만들어 낼 수가 있다. 결국 제어기 및 제어 신호 발생장치의 설계 문제는 그림12에 정의된 입력 신호를 이용하여 그림 5 의 TMS4161의 Timing조건을 만족하는 출력을 만들어내도록 하는 것이다. 이른 위하여, 그림12의 Dual-Pory RAM Controller (DPRC)의 입력과 출력간의 Timing Diagram 을 그려보면 그림13와 같게 된다.

지금까지 기술한 주소신호 발생기 및 요청 중재기를 이용하여 그림13의 Timing요건을 만족하게 하는 Dual-port RAM제어기의 전체적인 Block Diagram을 나타내면 그림14와 같다. 이제 전체적인 동작 수행에 필요한 Timing발생기 및 그림12의 제어기 (DPRC)의 설계에 대하여 기술하고자 한다.

TMS4161의 제어신호는 8203에서 발생이 되므로 모든 Cycle의 시작은 8203의 기동에서 비롯된다. 그런데 8203의 기동은 8203의 (RD)* 혹은 (WR)* 신호의 Activation에 의하여 이루어지므로 LINCYC이나 MEMCYC 모두가 이 신호

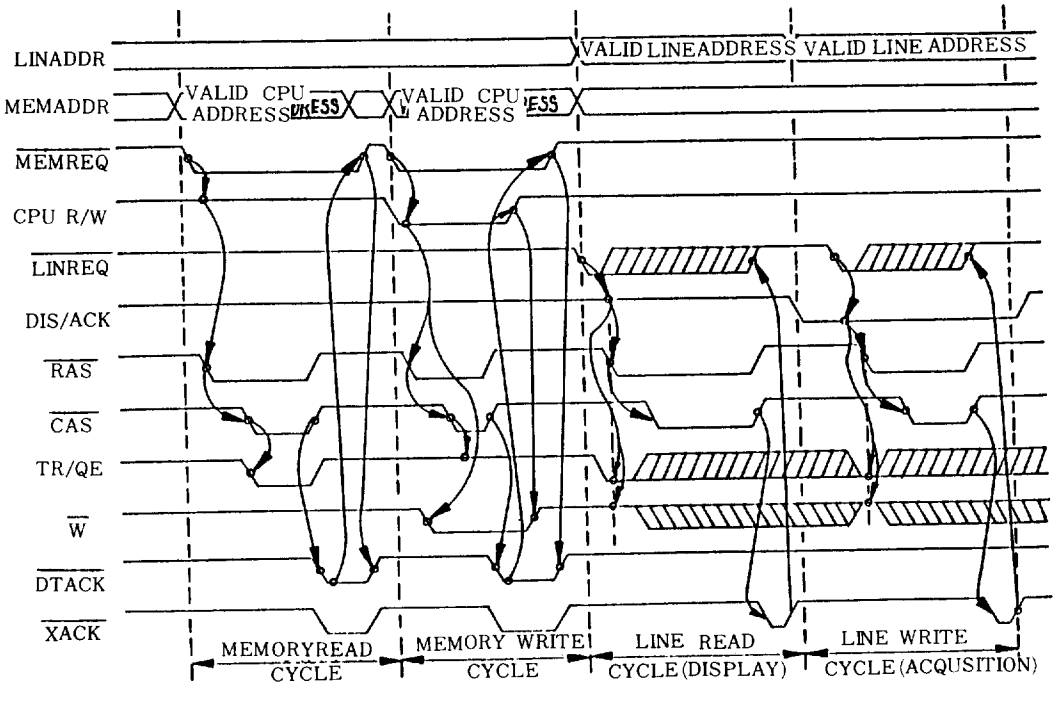


그림13. Dual-port RAM (TMS4161) 제어기의 Timing Diagram

를 사용하여 가동을 시켜야 한다. MEMCYC의 경우는 8203에서 출력되는 제어신호들 즉 (RAS)*, (RAS1)*, (CAS)* 및 (WE)*신호가 그림5 및 그림8(a)에서 보듯이 TMS4161의 Timing사양에 준하여 발생이 되므로 요구 증재기에서 MEMCYC임이 판정이 되면 8203을 기존의 DRAM 제어기 역할을 하게 하면 된다.

그러나 LINCYC의 경우는 Cycle의 기동을 위하여 8203에 (RD)* 혹은 (WR)* 신호를 Activation시켜야 하나 8203에서의 출력신호 (RAS0)*, (RAS1)*, (CAS)* 및 (WE)*만으로는 TMS4161의 제어신호를 모두 만들어 낼 수 없는 문제

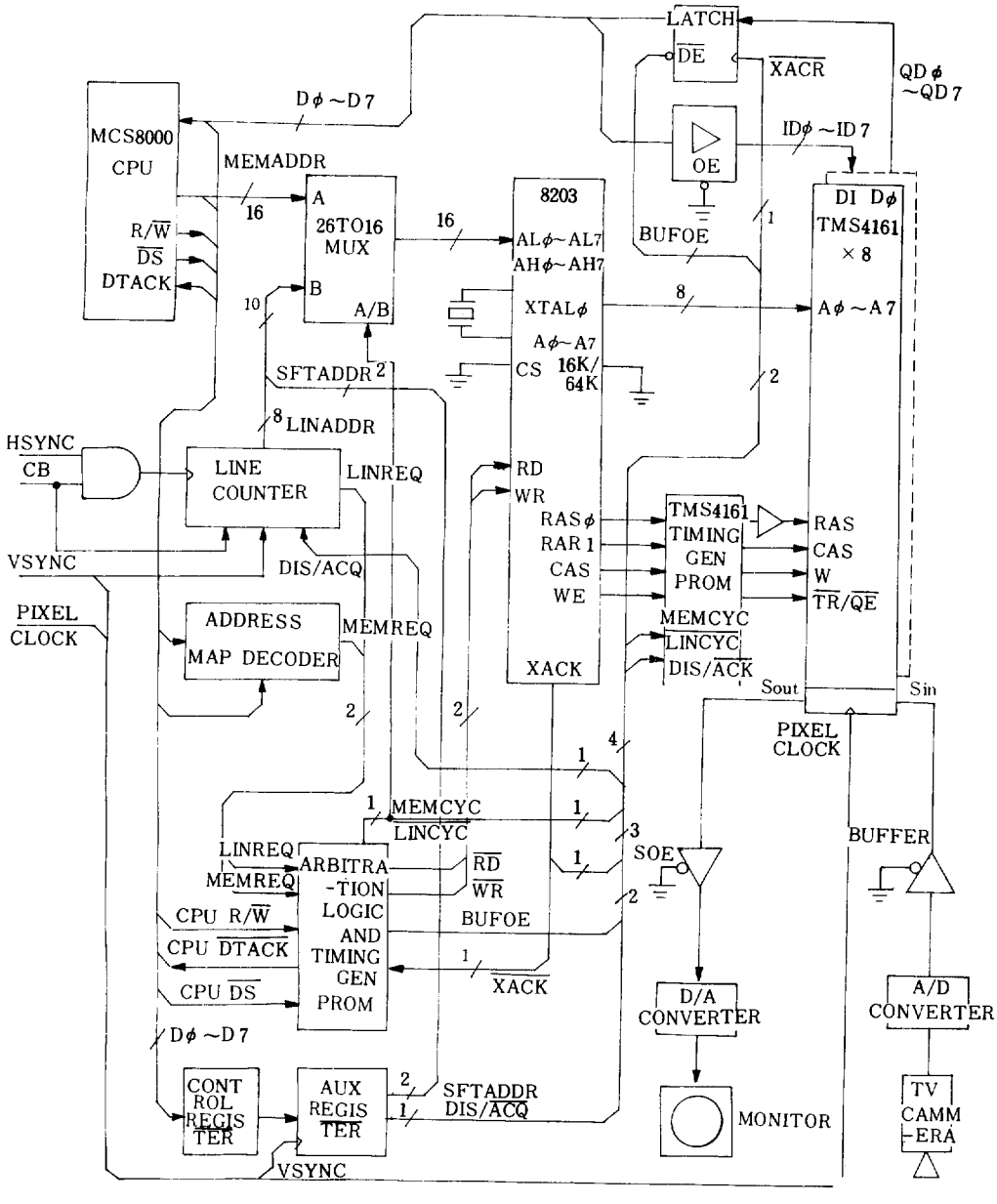


그림14. Dual-port RAM제어기

가 발생한다.

따라서 LINCYC의 기동을 8203의 (RD)*신호의 Activation에 의하여 만들고 8203의 출력신호들을 요구 중재기로 부터 발생된 LINCYC 신호에 의하여 변환시켜줄 필요가 있다. LINCY의 기동을 8203의 (WR)*의 Activation에 의하여 할 수 있으나 메모리 내용의 보호라는 면에서 보면 (RD)* 신호의 Activation이 바람직하다.

위의 설명한 바와 같이 8203의 (RD)* 및 (WR)*은 다음과 같은 Logic함수로 나타낼 수 있다.

$$(RD)^* = (MEMREQ)^* . (MEMCYC)^* . R/W + LINCYC^* . (EOLCYC)^{**}$$

$$(WR)^* = (MEMREQ)^* . (MEMCYC)^* . (R/W)^*$$

즉 8203의 RD* 신호는 MEMREQ이면서 요구중재기에서 MEMCYC임이 판명되고 CPU Read Cycle이거나 요구중재기에서 LINCYC임이 판명되고 LINCYC이 끝나지 않았을 경우 발생되게 한다. 그리고 8203의 (WR)* 신호는 MEMREQ이면서 요구중재기에서 MEMCYC을 허락하고, CPU Write Cycle일때 발생되게 한다.

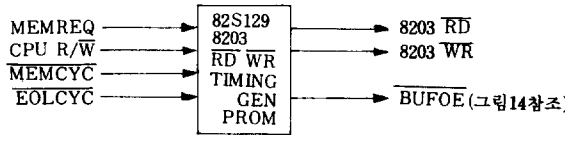
위와 같은 함수를 만족시키게하는 신호 발생을 중재기의 상태 및 출력 그리고 요구되는 신호를 이용하여 PROM을 사용하여 만들 수 있다. (그림15)

위와 같이 하여 8203의 (RD)*혹은 (WR)*신호의 기동에 의하여 요구중재기에서 선택된 Cycle이 시작이 되면 8203은 그림 8 (a)와 같이 Read 혹은 Write Cycle을 시작하여 (RAS0)*, (RAS1)*, (CAS)* 및 (WE)*을 발생하기 시작한다.

TMS4161의 RAS신호는 REFCYC, MEMCYC 및 LINCYC의 경우 모두 발생이 되어야 하므로 다음과 같은 함수로 나타낼 수 있다.

$$(RAS)^* \text{ of } TMS4161 = (RAS)^* + (RAS1)^*$$

그리고 CAS신호는 REFCYC에는 발생이 되지 말아야 하므로, $(CAS)^*$ of TMS4161 = $((RAS0)^* . (RAS1)^*)^* . ((CAS)^*$ of 8203)이어야 한다. 다시 말하면, 8203이 내부적인 Refresh Cycle의 수행중에는 (RAS)*과 (RAS1)*이 동시에



MEMREQ	CPU R/W	MEMCYC	EOLCYC	RD	WR	BUFOE
0	0	0	×	1	0	1
0	1	0	×	0	1	0
×	×	1	1	0	1	1
×	×	1	0	1	1	1

그림15. 8203 (RD)*및 (WR)* 신호 발생기 및 Truth Table

Active상태로 되므로 이때를 제외하고 8203의 (CAS)*를 TMS4161의 (CAS)*에 가해 주어야 한다.

또 TMS4161의 (W)*신호는 MEMCYC이면서 CPU Write일때 발생시켜야 하고 또한 LINCYC의 경우에서도 DIS/(ACQ)* = 0 즉 Acquition상태일때 발생이 되어야 하므로 (W)* of TMS4161 = (WE)*. (MEMCYC)* + ((RAS)* + (RAS1)*). (LINCYC)*. (DIS/ACQ)*가 된다. 여기서의 문제점은 Acquisition때, 그림13에서 보듯이 TMS4161의 (RAS)*보다 TMS4161의 (W)* 신호가 먼저 Active되어야 한다는 것이다. 그러나 그림16과 같이 TMS4161의 (W)* 및 (TR/QE)*의 Setup Time이 최소 0 (nsec)이므로 최종적인 (RAS)*를 Gate의 지연에 의하여 가해줌으로서 Setup Time을 만족시킬 수 있다.

그리고 이중 역할 신호인 TR/QE*는 LINCYC에는 RAS보다 먼저 그리고 CPU Rend Cycle일때는 CAS와 동일하면 되므로 다음과 같이 나타낼 수 있다.

$$(TR/QE)^* \text{ of } TMS4161 = (MECYC)^* . (WE)^{**} . (CAS)^* + ((RAS0)^* + (RAS1)^*) . (LINCYC)^*$$

여기서도 RAS보다 먼저 Active되어야 한다는 조건은 최종적인 RAS를 Gate지연에 의하여 가해줌으로써 해결할 수 있다. 그림17에 TMS 41

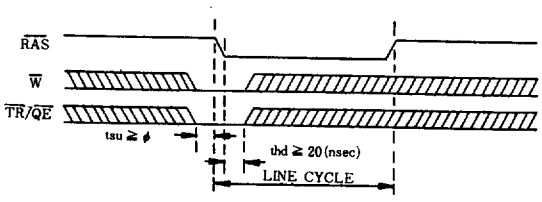
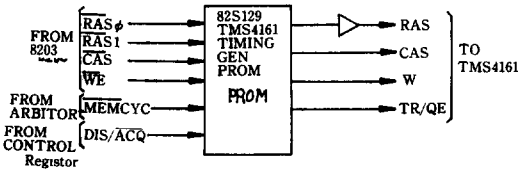


그림16. TMS4161의 Line Cycle시 (W)*, (TR)*/(QE)*의 Setup Time을 최소(nsec)이며 Hold Time은 최소20(nsec)이다.



RAS0	RAS1	CAS	WE	MEMCYC	DIS/ACQ	RAS	CAS	W	TR/QE	
φ	φ	X	X	X	X	φ	1	1	1	REFRESH CYCLE
1	φ	1	φ	φ	X	φ	1	φ	1	WRITE
1	φ	φ	φ	φ	×	φ	φ	φ	1	CYCLE
1	φ	1	1	φ	X	φ	1	1	1	READ
1	φ	φ	1	φ	X	φ	φ	1	φ	CYCLE
1	φ	1	1	1	φ	φ	1	φ	φ	ACQ
1	φ	φ	1	1	φ	φ	φ	φ	φ	CYCLE
1	φ	1	1	1	1	φ	1	1	φ	DIS
1	φ	φ	1	1	1	φ	φ	1	1	CYCLE

그림 17. TMS4161 제어신호 발생기 및 Truth Table

61의 (RAS)*, (CAS)*, (W)* 및 (TR)*/(QE)*의 함수를 만들어내는 제어신호 발생기 및 진리표를 나타내었다.

그림14의 제~, 레지스터(Control Register) 의 내용은 Image Memoy의 역할 즉 영상정보의 저장 혹은 Display를 제어하도록 CPU에 의하여 결정 될수 있으나 Frame 단위로 역할이 변화해 주기 위하여 보조 레지스터(Aux Kegister)를 이용하여 VSYNC(Vertical Sync)에 맞추어 실제적인 제어의 역할을 하게 한다.

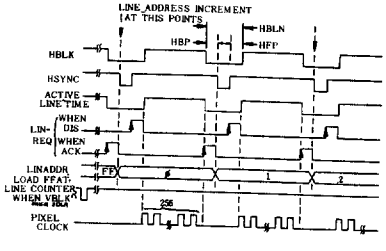
4.4 TMS4161을 이용한 영상입력장치의 전체적인 동작

본절에서는 지금부터의 설계의 결과로 나온 영상입력장치의 동작에 대하여 기술하고자 한다. (그림14 참조)

전체 동작의 기동은 MEMREQ 혹은 LINREQ 신호의 Activation에 의하여 시작이 되므로 이 신호들의 발생에 대하여 먼저 설명한다.

먼저 MEMREQ신호의 발생에 대하여 설명하면, CPU가 Image Memory영역을 사용하려고 할 때 그 주소신호를 Decoding하여 만든다. 즉 기존의 주기억 장치의 사용요청 방법과 동일하다.

다음에 LINREQ신호의 발생에 대하여 설명하면 다음과 같다. 영상정보의 저장 혹은 Display는 주사선(Scan Line)단위로 행하여 저야하므로 LINREQ신호는 유효주사시간 주에서는 수평주사의 시작 혹은 끝에서 발생이 되어야 한다. Acqusition때에는 TMS4161의 Shift Register)에 영상 정보가 모두 채워 졌을때 Memory Array로 전달하여야 하므로 수평 주사의 LINREQ가 발생되어야 하나 Display때에는 수평주사의 시작 전에 먼저 Memory Array로 부터 Shift Register에 옮겨 놓아야 하므로 수평 주사의 처음에 LINREQ가 발생되어야 한다. (그림18) 위와 같이



HBLK : Horizontal Blanking Interval
HBP : Horizontal Back Porch
HFP : Horizontal Front Porch

그림18. VIDEO신호의 Timing 및 LINREQ 신호의 발생. Display때와 Acquisition때에 LINREQ신호의 발생이 달라야 하며 Line Address의 변화는 두 신호의 사이에서 행하여져야 한다.

Acquition 경우와 Display 경우를 분리하여 LINREQ를 만들어 내야 하나 Line Address가 변하지 않고 Shift Register에 Pixel Clock이 가해지지 않는 구간에서는 Shift Register의 내용이 변하지 않는 것을 이용하면 HBLK (Herizontal Blank) 구간동안에 LINREQ를 두번 발생하게 하고 수평주사 시간의 끝에나오는 LINREQ신호만 Acquition 혹은 Display에 따라 제어하면 된다.

그림19에 이와 같은 방법을 이용한 LINREQ 신호발생기의 Timing Diagram, Circuit Diagram 및 진리표를 나타내었다. HBLK구간에서 LINREQ신호를 두번 발생하게 하고 만일 Acquisition 상태이면 TMS4161 제어신호 발생기의 입력으로 작용하는 DIS/ACQ신호를 Toggling하게 한다.

따라서 각 수평주사구간의 처음에는 항상 TMS4161의 Memory Array로 부터 Shift Register로 Data이동이 이루어지고 수평주사구간의 끝에서는 그 Data를 다시 읽거나 혹은 A/D 변환기로 부터 Shift Register로 병렬 입력된 정보를 Memory Array에 쓰게 된다. 그러므로 Acquisition의 경우에 Sout (Shitf Out) deta는 그전에 Memory Array에 들어 있는 내용D (f) (n+1)이 되고 Sin (Shift in) data는 현재 A/D 변환기로 부터 들어오는 내용D (f) (n+1)이 된다. 따라서 Acqusition동안에도 Memory 내용을 TV Monitor에 그 전 Frame의 내용을 Display할 수 있어 화면의 깜빡거림 (Flickering)이 전혀 없게 된다.

이와 같이 MEMREQ 신호 및 LINREQ신호가 발생되면 요구 중재기 (Request Arbiter) 에서는 4.2절에서 설명한 바와 같이 MEMCYC 혹은 LINCYC 중 어느것을 수행해 주어야 할지를 결정한다. 그 결정에 의하여 그림15의 Timing 발생기는 8203에 Cycle시작을 요구하고 8203은 그림8에 주어진 Timing에 의하여 제어 신호인 (RAS0)*, (RAS1)*, (CAS)* 및 (W)* 신호를 발생하게 되며 TMS4161제어 신호 발생 PROM에서는 위의 네 가지 신호와 요구 중재기 에서 판별된 MEMCYC/(LINCYC)* 신호를 이용하여 주어진 요구에 맞는 TMS4161 제어 신호를 만들어 낸다. 그리고 MEMCYC, 즉 CPU가영상 Memory 내용을 Access할 경우는 TMS 4161이 출력 Data는 Latch를 통하여 CPU로 하여금 읽어가는 동안에 Data의 변화를 막아주고 입력 Data는 Buffer를 통하여 TMS4161의 입력 Data단자에 가해진다.

LINCYC, 즉 영상 정보의 입출력의 경우는, 출력일 경우에는 D/A 변환기를 통하여 TV Monitor에 Display되고, 입력의 경우는 A/D 변환기를 통하여 들어온 정보가 Memory Array의 한 Row에 저장이 된다.

지금부터는 위와 같이 구성한 메모리가 CPU의 주기억장치와 같이 사용될수 있다는 것을 설명한다. 그림19에서 보는 바와 같이 LINREQ가 들어와서 LINCYC을 수행하는 횟수는 한 수평주사 시간중에서 두번이다. 따라서 그 두번의 시간 구간을 제외하고는 CPU가 항상 영상 Memery의 사용이 가능하다. TMS4161의 LINCYC 수행시간은 약 900 (nsec) 정도 이므로 한 수평주사 시간중 1.8 (μsec)의 시간이 LINCYC에 할당이 된다. 따라서 한 frame당 유효 수평주사선의 갯수를 256개로 할 경우 한 Frame에서 LINCYC에 의하여 사용되는 시간은

$$256 \times 1.8 = 461\ (\mu\text{sec})$$

가 된다. 60Hz의 Refresh Rate를 갖는 영상 시스템의 경우 한 Frame에 걸리는 시간은 1/60 =16.667 (msec) 이므로 결국 한 Frame시간 중에서

$$\frac{0.461}{16.667} \times 100 = 2.8\ (\%)$$

만이 LINCYC에 사용이 되고 나머지 97.2 (%)의 시간을 CPU가 사용할 수 있게 된다. 또한 CPU가 영상 메모리를 사용하려고 할 때 현재 LINCYC이 진행중에 있는가의 여부를 판별할 필요 없이 직접 사용 요청을 해도 요구 중재기에서 그에 대한 중재를 행하여 주므로, LINCYC이 진행중인 경우에만 LINCYC기간 만큼 사용이 지연될 뿐, 결국 CPU의 주기억 장치와 같은 메모리로 보고 사용할 수 있다.

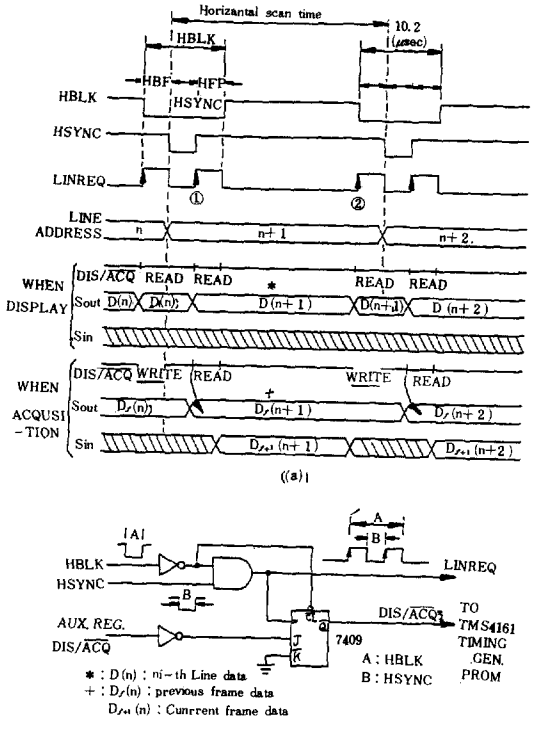
이와 같이 CPU가 영상 메모리를 마치 주 기억 장치와 같이 사용할 수 있다는 장점을 이용

하여 해상도를 높이면 Graphic용도로도 충분히 이용할 수도 있다.

5. 실험 결과

본 논문에서 제시한 이중 입출력 메모리(Dual-Port RAM)를 이용한 영상 입력 장치의 타당성을 입증하기 위하여 주소신호 발생기 및 요구중재기를 실제 제작하여 실험하였다.

CPU로서는 16/32bit 마이크로 프로세서인 M



	HBLK	HSYNC	AUX.REG. DIS/ACQ	CLEAR	$\overline{Q_n}$	$\overline{Q_{n+1}}$ (DIS/ACQ)
WHEN DISPLAY	L	H	H	H	H	H
WHEN ACQUSITION	L	H	L	H	H → L (TOGGLE)	
	L	H	L	L	L → H (TOGGLE)	

(c)

그림19. LINREQ신호 발생기

C68000을 이용하였고(7)시스템 버스(System bus)로는 16/32bit마이크로 프로세서에 적합한 VME bus(2)를 택하였다.

본 논문에서 제시한 영상 입력 장치가 이중입출력이 가능하므로 실시간 처리가 가능하다는 점을 충분히 활용키 위하여 영상 데이타 전용의 버스 3개를 구성하여 확장성이 가능하도록 전체 실험 장치를 구현하였다. (그림20)

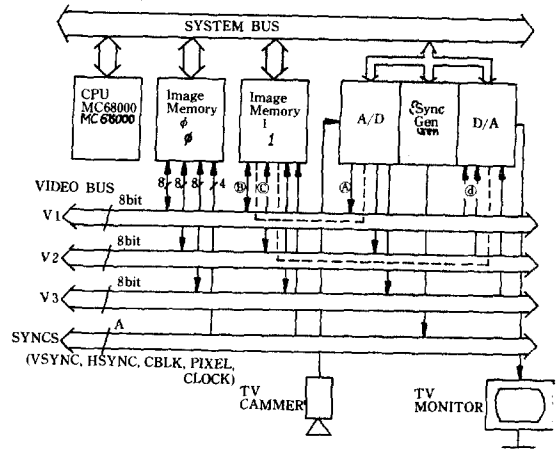


그림20. 실험 장치의 Block Diagram 각 메모리 장치 및 A/D, D/A 변환기에는 24 to 8 입력 Mux 및 8 to (3×8) 출력 Demux가 붙어있다.

A/D 변환기는 RCA사의 CA3308 (66 (nsec) 변환 시간)을 사용하였으며 D/A변환기는 TRW사의 TDC1016 (50 (nsec) 변환시간)을 사용하였다. 동기 신호 발생기로는 NEC사의 uPD 7220을 사용하였다. TV카메라는 외부 동기 구동이 가능한 CCTV를 사용하였으며 12" Moniter에 실험 결과를 Display하게 하였다.

그림20. 에서 한장의 영상을 얻기 위하 여는 점선 부분에 해당하는 Data Path만 형성하여주면 된다. 즉, A/D 변환기로부터 Frame Rate로 발생되는 영상 Data를 V1-Bus에 내보내고 (A), 영상 메모리 1 (Image Memory 1)에서 V1-Bus로 그 정보를 읽어 들이면서 (B), 다시 V2-Bus로 동시에 내보낸다.(C) 그러면 D/A 변환기는 V2-Bus에서 읽어들여서 TV Monitor 에

Display하게 된다. 이러한 일련의 과정이 CPU 사용에는 관계없이 발생이 된다.

이러한 Bus구조를 더욱 확장하여 패턴 매칭의 한 방법인 영상 메모리간의 연산 기능도 충분히 가능하리라고 사료된다.

6. 결 론

본 논문에서는 이중입출력 메모리(Dual-Port RAM)를 이용한 영상 입력장치(Image Memory)의 설계 및 그 제어 신호 발생기에 대하여 논하였다.

이중 입출력 메모리 소자인 TMS4161은 기존의 표준 64K×1 DRAM Port와 256bit의 내부적 Shift Register와 연결된 Serial Port가 있어서, 실시간 영상 처리 및 그래픽 용으로 사용하기에 적합하나, 그 사용에 있어서 가장 어려운 문제로 제안(3)된 주소 신호 발생기 및 요구 중재기에 대한 해결 방안을 제시하였다.

또한 서로 독립적인 두개의 입출력 장치가 있다는 장점을 이용하여 하드웨어에 의한 실시간 처리도 가능한 구조로 쉽게 확장할 수 있어서 소프트 웨어에 의한 실시간 처리로 가능하리라 사료된다.

앞으로는 512×512×8의 영상 메모리 구조뿐만 아니라 1024×1024×8의 영상메모리 구조에 대하여 더욱 연구할 필요가 있다고 본다.

참 고 문 헌

1) Mary C. Whitton, "Memory Design for Rester Graphic Display", IEEE CG & A, March, 1984.
2) "VMEbus Specification", Motorola, Inc., Manual.
3) Ray Pickham et al., "Video RAM excels at fast graphics", Electronic Design, Aug., 1983.
4) Richard Matick, "All points addressible raster display memory", IBM J. RES. Develop., Vol. 28, No. 4, July, 1984.
5) "TI Memory Component", Texas Ins., Manual.
6) "Handbook of Microcontrollers", Intel. Inc., Manual.
7) "MVME110-1MC68000Microcomputer User's Manual", Motorola Inc.