(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl. [6]
G06F 13/16

(45) 공고일자 2002년04월06일
(11) 등록번호 10-0319331
(24) 등록일자 2001년12월18일

(21) 출원번호 10-1994-0010356
(22) 출원일자 1994년05월12일

(65) 공개번호 특1994-0026744
(43) 공개일자 1994년12월09일

(30) 우선권주장 93-112841 1993년05월14일 일본(JP)

(73) 특허권자
소니 컴퓨터 엔터테인먼트 인코포레이티드
도쿠나까 테루히사
일본 도쿄도 미나도구 아가사가 7-1-1

(72) 발명자
야마모또야스유끼
일본도오꾜도시나가와꾸기다시나가와6죠메7반35고소니가부시끼가이샤나이
이시바시도시야
일본도오꾜도시나가와꾸기다시나가와6죠메7반35고소니가부시끼가이샤나이

(74) 대리인
이병호
이병호

심사관 : 오흥수

(54) 버스제어장치및버스제어방법

요약

장치 전체를 대형화 고가화를 하는 일 없이 중앙 연산 처리 장치와 주변 장치와의 사이의 액세스 타이밍을 제어할 수가 있도록 한다.

타이밍 테이블(제 3 도)에 기술되어 있는 주변 장치(ROM(5), 백업 메모리 (6), 디바이스(7, 8))의 동작 타이밍에 의거해서 중앙 연산 처리 장치와 주변 장치와의 사이의 액세스 타이밍을 제어한다.

대표도

도 1

명세서

도면의 간단한 설명

제 1 도는 본 발명의 버스 제어 장치를 적용한 컴퓨터의 한 실시예의 구성을 도시하는 블록도.

제 2 도는 제 1 도의 실시예에 있어서 게이트웨이(4)의 보다 상세한 블록도.

제 3 도는 타이밍 테이블을 도시하는 도면.

제 4도는 버스 폭 테이블을 도시하는 도면.

제 5a 내지 5i 도는 디바이스(7)로부터 CPU(1)로 데이터의 판독이 행해질 때의 동작을 설명하는 타이밍 차트.

제 6a 내지 6i 도는 CPU(1)로부터 디바이스(8)로부터 데이터의 기입이 행해질 때의 동작을 설명하는 타이밍차트.

제 7a 내지 7i 도는 ROM(5)에서 CPU(1)로 데이터의 판독이 행해질 때의 동작을 설명하는 타이밍차트.

제 8a 내지 8i 도는 CPU(1)로부터 디바이스(7)로 데이터의 기입이 행해질 때의 동작을 설명하는 타이밍차트.

제 9 도는 게이트웨이(4)의 동작을 설명하는 플로차트.

제 10 도는 제 9 도의 플로차트의 단계 S13 의 데이터의 판독 처리의 보다 상세한 것을 설명하는 플로차트.

도 11 도는 제 10 도의 플로차트의 단계 S21 의 복원 처리의 보다 상세한 것을 설명하는 플로차트.

제 12 도는 제 10 도의 플로차트의 단계 S29 의 다음의 어드레스의 출력 처리의 보다 상세한 설명을 하는 플로차트.

제 13 도는 제 9 도의 플로차트의 단계 S12 의 데이터의 기록 처리의 보다 상세한 것을 설명하는 플로차트.

제 14 도는 제 13 도의 플로차트의 단계 S60 의 다음의 어드레스, 데이터의 출력 처리의 보다 상세한 것을 설명하는 플로차트.

* 도면의 주요부분에 대한 부호의 설명

4 : 게이트웨이 11 : 데이터 버퍼

12a 내지 12d : 다중기 13 : 타이밍 테이블 기억부

14 : 버스 폭 테이블 기억부 15 : 카운터

16 : 컨트롤러 17 : 메인 버스 어드레스 디코더

18 : 어드레스 테이블 기억부 19 : 서브 버스 어드레스 디코더

20 : 상태 버퍼 21 : 어드레스 버퍼

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

산업상의 이용분야

본 발명은 예를 들면 컴퓨터 등에 사용하기에 가장 적합한 버스 제어 장치에 관한 것이다.

종래의 기술

종래의 컴퓨터에 있어서는 그것의 CPU 의 버스 액세스 속도 보다 저속의 주변 장치(예를 들면 인터페이스 보드 등)를 사용하는 경우, 그것의 주변 장치의 버스 액세스 타이밍에 맞추어서 CPU 의 버스 액세스 타이밍을 지연시키면 장치 전체의 처리 속도가 저하하기 때문에, 이를테면 대기 컨트롤러(wait controller) 등을 설치해 CPU 가 출력한 판독 신호나 기록 신호 등을 주변 장치에 공급하는 타이밍을 조정하도록 되어 있다.

또한 종래의 컴퓨터에서는 그것의 CPU의 버스 폭과 상이한 버스 폭의 주변 장치를 사용하는 경우, 즉 예를 들면 32 비트의 버스 폭의 CPU 에 대해 8 비트의 버스 폭의 주변 장치를 사용하는 경우 데이터의 팩/언팩을 하는 전용의 하드웨어 기구를 설치하고 CPU 가 출력한 32 비트의 데이터를 8 비트의 데이터로 4 분할해서 (언팩) 주변 장치에 공급하고 또한 주변 장치가 출력한 8 비트의 데이터를 4 회 입력하여 32 비트의 데이터로 하여 (팩), CPU 에 공급하도록 되어 있다.

발명이 해결하려고 하는 과제

그런데 상술하는 대기 컨트롤러는 컴퓨터에 접속하는 주변 장치에 맞추어서 개별로 설계하지 않으면 안되고 장치가 고가가 되는 과제가 있었다.

또한 주변 장치에 따라서는 CPU 가 주변 장치로부터 데이터를 불러들이는 경우 CPU 의 데이터의 판독이 종료해도 얼마동안 데이터를 계속 출력을 하고 있고 즉 플로팅 타임이 긴 것이 있다. 또 주변 장치에 따라서는 CPU 가 주변 장치에 데이터를 기입하는 경우 CPU 의 데이터의 기입 신호의 출력이 종료해도 얼마동안 데이터 버스 상에 데이터를 유지할 필요가 있는 즉 유지 시간(hold time)이 긴 것이 있다.

이와 같은 플로팅 타임이나 유지 시간이 긴 주변장치에 대해서는 상술하는 대기 컨트롤러만으로는 동작 타이밍을 조정할 수 없고 그 주변 장치에 하드웨어적인 연구를 실시하지 않으면 안되었다.

따라서 장치가 대형화, 고가가 되는 과제가 있었다.

또한 상술하는 팩/언팩을 행하는 하드웨어 기구도 대기 컨트롤러와 같이 컴퓨터에 접속하는 주변 장치에 맞추어서 예를 들면 팩/언팩의 트리거 발생 논리 등을 개별로 설계하지 않으면 안되고 장치가 고가가 되는 과제가 있었다.

다시 예를 들면 상술하는 바와 같이 32 비트의 버스 폭의 CPU 에 대해 8 비트의 버스 폭의 주변 장치를 사용하는 경우 CPU 가 출력한 32 비트의 데이터를 언팩하고 4 개의 8 비트의 데이터로 하여 이들을 주변 장치에 기입할 때 그것의 주변장치가 예를 들면 메모리이면 데이터를 기입하는 어드레스를 1 개씩 4 회 증분할 필요가 있으나 주변 장치가 예를 들면 FIFO 형의 버퍼 등을 갖춘 데이터의 읽고 쓰기가 고정 어드레스로 행해지는 것이라면 동일(고정) 어드레스에 대해 4 개의 8 비트의 데이터를 기입하도록 할 필요가 있다.

여기에서 본 명세서 중에 있어서는 CPU 와 버스 폭이 다른 주변 장치에 대해서 데이터를 읽고 쓰는 경우에 어드레스가 상술하는 바와 같이 증분되거나 고정이거나 하는 것을 어드레스 형식이라 한다.

그러나 상술하는 하드웨어 기구는 주변 장치의 어드레스 형식을 고려해서 주변 장치에 대해서 데이터를 읽고 쓰는 어드레스를 지정하도록 되어 있지 않았다.

따라서 예를 들면 응용 프로그램 등은 주변 장치의 어드레스 형식을 고려해서 설계, 제작하지 않으면 안되므로 번거로웠다.

본 발명은 이와 같은 상황을 감안해서 이루어진 것이고 장치를 대형화, 고가가 되는 일없이 CPU 와 주변 장치 사이의 데이터의 주고받음을 제어할 수가 있도록하는 것이다.

과제를 해결하기 위한 수단

본 발명의 버스 제어 장치는 예를 들면 CPU(1) 등의 중앙 연산 처리 장치가 접속된 메인 버스(9)와 예를 들면 ROM(5)나 백업 메모리(6), 디바이스(7, 8)등의 주변 장치가 접속된 서브 버스(10)와의 사이에 접속되는 버스 제어 장치로서 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 동작 타이밍을 기억하고 있는 기억 수단으로서의 타이밍 테이블 기억부(13)를 구비하고 타이밍 테이블 기억부(13)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 동작 타이밍에 의거해서 CPU(1)과 ROM(5), 백업 메모리(6), 디바이스(7, 8)와의 사이의 액세스 타이밍을 제어하는 것을 특징으로 한다.

본 발명의 버스 제어 장치는 타이밍 테이블 기억부(13)에 기억된 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 동작 타이밍이 고쳐 쓸 수 있는 것을 특징으로 한다.

본 발명의 버스 제어 장치는 타이밍 테이블 기억부(13)에는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 최소한 유지 시간 또는 복원시간이 기억되어 있는 것을 특징으로 한다.

본 발명의 버스 제어 장치는 예를 들면 CPU(1) 등의 중앙 연산 처리 장치가 접속된 메인 버스(9)와 예를 들면 ROM(5)나 백업 메모리(6), 디바이스(7, 8)등의 주변 장치가 접속된 서브 버스(10)와의 사이에 접속되는 버스 제어 장치로서 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 버스 폭과 어드레스 형식을 기억하고 있는 기억 수단으로서의 버스 폭 테이블 기억부(14)와 CPU(1) 또는 ROM(5), 백업 메모리(6), 디바이스(7, 8)가 출력하는 데이터를 래치하는 데이터 래치 수단으로서의 데이터 버퍼(11)와 CPU(1)이 ROM(5), 백업 메모리(6), 디바이스(7, 8)로 출력하는 어드레스를 래치하는 어드레스 래치 수단으로서의 어드레스 버퍼(21)와 버스 폭테이블 기억부(14)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 버스 폭에 의거해서 데이터 버퍼(11)에 래치된 데이터의 팩/언팩을 제어함과 함께 버스 폭 테이블 기억부(14)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 어드레스 형식에 의거해서 어드레스 버퍼(21)에 래치된 어드레스를 제어하는 제어 수단으로서의 컨트롤러(16)를 구비하는 것을 특징으로 한다.

본 발명의 버스 제어 장치는 버스 폭 테이블 기억부(14)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7,8)의 버스 폭과 어드레스 형식이 고쳐 쓸 수 있는 것을 특징으로 한다.

작 용

본 발명의 버스 제어 장치에 있어서는 타이밍 테이블 기억부(13)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 동작 타이밍에 의거해서 CPU(1)와 ROM(5), 백업 메모리(6), 디바이스(7, 8)와의 사이의 액세스 타이밍을 제어한다. 따라서, CPU(1)와 ROM(5), 백업 메모리(6), 디바이스(7, 8)와의 사이의 액세스 타이밍을 조정하는 예를 들면 대기 컨트롤러 등의 하드웨어를 주변 장치마다 설치할 필요가 없어지므로 장치 전체를 대형화, 고가화 하는 일이 없이 CPU(1)와 ROM(5), 백업 메모리(6), 디바이스(7, 8)와의 사이의 액세스 타이밍을 제어할 수가 있다.

본 발명의 버스 제어 장치에 있어서는 타이밍 테이블 기억부(13)에 기억된 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 동작 타이밍이 고쳐 쓸 수 있으므로 예를 들면 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 사양이 변경되거나 새로운 주변 장치가 추가되어도 장치가 실질적으로 사용이 불가능해지는 것이 방지된다.

본 발명의 버스 제어 장치에 있어서는 타이밍 테이블 기억부(13)에는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 최소한 유지 시간 또는 복원 시간이 기억되어 있으므로 유지 시간 또는 복원 시간에 의거한 타이밍 제어를 할 수가 있다.

본 발명의 버스 제어 장치에 있어서는 데이터 버퍼(11)에 의해 CPU(1) 또는 ROM(5), 백업 메모리(6), 디바이스(7, 8)가 출력하는 데이터를 래치하고, 어드레스 버퍼(21)에 의해 CPU(1)가 ROM(5), 백업 메모리(6), 디바이스(7, 8)에 출력하는 어드레스를 래치한다. 그래서 버스 폭 테이블 기억부(14)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 버스 폭에 의거해서 데이터 버퍼(11)에 래치된 데이터의 팩/언팩을 제어함과 함께 버스 폭 테이블 기억부(14)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 어드레스 형식에 의거해서 어드레스 버퍼(21)에 래치된 어드레스를 제어한다. 따라서 데이터의 팩/언팩 및 어드레스 형식을 고려하는 일없이 응용 프로그램을 설계할 수가 있게 되고 범용성이 높은 응용 프로그램을 제공할 수가 있게 된다.

본 발명의 버스 제어 장치에 있어서는 버스 폭 테이블 기억부(14)에 기억되어 있는 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 버스 폭과 어드레스 형식이 고쳐 쓸 수가 있으므로 예를 들면 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 사양이 변경되건 새로운 주변 장치가 추가되어도 장치가 실질적으로 사용이 불가능해지는 일이 방지된다.

실시예

제 1 도는 본 발명의 버스 제어 장치를 적용한 게이트웨이(4)가 설치된 컴퓨터의 한 실시예의 구성을 도시하는 블록도이다. CPU(1)는 예를 들면 32 비트의 CPU 로 ROM(5)에 기억되어 있는 시스템 프로그램이나 RAM(2)등에 기억된 응용 프로그램에 따라서 소정의 처리를 한다.

CPU(1), RAM(2) 및 디바이스(3)는 32 비트의 고속 버스로서의 메인 버스(9)에 접속되고 CPU(1)와 RAM(2) 또는 디바이스(3)와의 사이에는 메인 버스(9)를 거쳐서 고속으로 32 비트의 데이터가 주고받도록 되어 있다.

ROM(5)은 예를 들면 8 비트 폭의 ROM으로 시스템 프로그램을 기억하고 있다. 백업 메모리(메모리)(6)는 예를 들면 16 비트 폭의 메모리에서 모터 등의 백업을 기억한다. 디바이스(7)는 예를 들면 8 비트 폭의 디바이스로 음성 처리나 화상 처리 등의 소정의 처리를 한다. 디바이스(8)는 예를 들면 16 비트 폭의 입출력 인터페이스이다.

ROM(5), 메모리(6), 디바이스(7 및 8)는 어느 것이나 그것의 동작 속도가 CPU(1) 보다 저속으로 예를 들면 16 비트의 범용 버스로서의 서브 버스(10)에 접속되어 있다. 단 8 비트 폭의 ROM(5)와 디바이스(7)는 16 비트의 서브 버스(10)의 하위 8 비트에 접속되어 있다.

또한 도시하지 않았으나 메인 버스(9) 및 버스(10)는 데이터 버스 어드레스 버스 및 제어 버스로 구성이 되어 있다.

게이트웨이(4)는 메인 버스(9)와 서브 버스(10)와의 사이에 접속되고, CPU(1)와 ROM(5), 메모리(6), 디바이스(7 또는 8)와의 사이의 데이터의 주고받기를 제어한다.

즉, 게이트웨이(4)는 예를 들면 제 2 도에 도시하는 바와 같이 데이터 버퍼 (11), 다중화기(MUX)(12a 내지 12d) 타이밍 테이블 기억부(13), 버스 폭 테이블 기억부(14), 카운터(15), 컨트롤러(16), 메인 버스 어드레스 디코더(17), 어드레스 테이블 기억부(18), 서브 버스 어드레스 디코더(19), 상태 버퍼(20), 어드레스 버퍼(21)로 구성이 된다.

또한 제 2 도에 있어서 점선은 제어 신호 선을 의미한다.

데이터 버퍼(11)는 컨트롤러(16)에 제어되고 CPU(1)에서 메인 버스(9)를 거쳐서 공급되는 32 비트의 데이터를 래치하여 그것의 상위로부터 8 비트마다의 4개의 데이터를 MUX(12a 내지 12d)에 각각 출력한다. 또한 데이터 버퍼(11)는 8 비트의 다중화기(12a 내지 12d)로부터의 데이터를 래치하고 32 비트의 데이터로서 메인 버스(9)에 출력한다.

각각 8 비트 폭으로 구성되어 있는 MUX(12a 내지 12d)는 컨트롤러(16)에 제어되고 데이터 버퍼(11)로부터의 4 개의 8 비트의 데이터를 각각 래치하여 2 개의 16 비트의 데이터, 또는 4 개의 8 비트의 데이터로서 서브 버스(10)로 출력한다. 또한 MUX(12a 내지 12d)는 서브 버스(10)로부터의 2 개의 16 비트의 데이터 또는 4 개의 8 비트의 데이터를 래치하고 32 비트의 데이터로서 데이터 버퍼(11)로 출력한다.

타이밍 테이블 기억부(13)에는 서브 버스(10)에 접속된 ROM(5), 메모리(6), 디바이스(7 또는 8) 각각의 예를 들면 판독 지연시간, 기록 지연 시간, 유지 시간, 플로팅 시간 및 복원 시간이 기술된 제 3 도에 도시하는 바와 같은 타이밍 테이블이 기억되어 있다.

또한, 판독 지연시간 또는 기록 지연시간과는 각각 데이터가 판독 또는 기록될 때의 액세스 타임을 의미하고 유지 시간과는 기록 신호의 출력이 종료해도 버스위에 데이터를 유지할 필요가 있는 시간을 의미한다. 또한, 플로팅 타임이란 데이터의 판독이 종료해도 버스 위에 데이터를 보존할 필요가 있는 시간을 의미하고 복원 시간이란 동일한 주변 장치에 연속해서 액세스할 때에 필요한 간격(1번째의 액세스가 종료하고서부터 다음의 액세스까지에 설치할 필요가 있는 시간 간격)을 의미한다.

버스 폭 테이블 기억부(14)에는 서브 버스(10)에 접속된 ROM(5), 메모리(6), 디바이스(7 또는 8)가 처리하는 데이터 길이(이하 버스 폭이라 함)와 그것의 어드레스 형식이 기술된 제 4 도에 도시하는 바와 같은 버스 폭 테이블이 기억되어 있다.

또한 제 4 도의 버스 폭 테이블에 있어서 어드레스 형식이 고정으로 되어 있는 주변 장치(본 실시예에 있어서는 디바이스(7 및 8))는 예를 들면 FIFO 형의 데이터 버퍼를 내장하고 있고 언팩되어서 복수로 분할된 데이터의 판독 기록 때에 소정의 동일 어드레스(고정 어드레스)를 지정할 필요가 있는 것이다. 또한, 어드레스 형식이 가변으로 되어 있는 주변 장치(본 실시예에 있어서는 ROM(5) 및 메모리 (6))는 언팩되어서 복수로 분할된 데이터의 읽고 쓰기 때에, 분할된 데이터의 한번의 읽고 쓰기 후, 다음의 데이터를 읽고 쓰는데에 어드레스를 증분(또는 감산)해서 지정할 필요가 있는 것이다.

카운터(15)는 도시하지 않은 클록 발생기로부터 메인 버스(9) 및 컨트롤러 (16)를 거쳐서 공급되는 시스템 클록을 카운트한다.

컨트롤러(16)는 타이밍 테이블 기억부(13)에 기억된 타이밍 테이블(제 3 도)에 의거해서 CPU(1)로부터 메인 버스(9)를 거쳐서 공급되는 판독 신호(Read) 또는 기록 신호(Write)의 타이밍을 바꾸어, SRead 신호 또는 SWrite 신호로서 서브 버스 (10)로 출력한다. 또 컨트롤러(16)는 타이밍 테이블 기억부(13)에 기억된 타이밍 테이블(제 3 도)에 의거해서 소정의 타이밍으로 후술하는 DataOk 신호를 출력한다.

다시 컨트롤러(16)는 버스 폭 테이블 기억부(14)에 기억된 버스 폭 테이블 (제 4 도)에 기술된 버스 폭 및 CPU(1)에서 메인 버스(9)를 거쳐서 공급되는 AccessType(CUP1이 취급하는 데이터 길이)에 의거해서 데이터 버퍼(11)를 제어하고 데이터를 팩/언팩 시킨다. 또한, 컨트롤러(16)는 버스 폭 테이블 기억부(14)에 기억된 버스 폭 테이블(제 4 도)에 기술된 어드레스 형식에 의거해서 어드레스 버퍼 (21)를 제어한다.

메인 버스 어드레스 디코더(17)는 CPU(1)로부터 메인 버스(9)를 거쳐서 출력된 어드레스를 디코드하여 그것의 디코드 결과와 어드레스 테이블 기억부(18)에 기억되어 있는 어드레스 디코드 테이블을 비교한다. 그래서 그것의 비교 결과에서 CPU(1)이 액세스하려고 하는 주변 장치가 메인 버스(9)에 접속된 RAM(2) 또는 디바이스(3)인가의 여부를 판정하고 그것이 RAM(2) 또는 디바이스(3)라고 판정한 경우에는 메인 버스(9)를 거쳐서 RAM(2) 또는 디바이스(3)에 칩 선택 신호를 출력한다.

어드레스 테이블 기억부(18)는 RAM(2), 디바이스(3), ROM(5), 메인(6), 디바이스(7 또는 8)에 각각 할당된 어드레스의 디코드 결과가 기술된 어드레스 디코드 테이블을 기억하고 있다.

서브 버스 어드레스 디코더(19)는 CPU(1)로부터 메인 버스(9) 및 어드레스 버퍼(21)를 거쳐서 출력된 어드레스 디코드하고 그 디코드 결과와 어드레스 테이블 기억부(18)의 어드레스 디코드 테이블에 기술되어 있는 RAM(2), 디바이스(3), ROM(5), 메모리(6), 디바이스(7 또는 8)에 각각 할당된 어드레스의 디코드 결과를 비교한다. 그래서 그것의 비교 결과로부터 CPU(1)이 액세스하려고 하는 주변 장치가 서브 버스(10)에 접속된 ROM(5), 메모리(6), 디바이스(7 또는 8)인가의 여부를 판정하고 그것이 서브 버스(10)에 접속된 주변 장치 중의 어떤 것(ROM(5), 메모리 (6), 디바이스(7 및 8) 중의 어떤 것)으로 판정된 경우에는 그것의 주변 장치에 서브 버스(10)를 거쳐서 칩 선택 신호를 출력한다.

상태 버퍼(20)는 메인 버스 어드레스 디코더(17) 및 서브 버스 어드레스 디코더(19)중의 어떤 것이 최후로 칩 선택 신호를 출력한 주변 장치 즉 CPU(1)가 최후로 액세스한 주변 장치(주변 장치에 할당된 어드레스의 디코드 결과)를 기억한다. 다시 상태 버퍼(20)는 그것의 기억치와 다음에 서브 버스 어드레스 디코더 (19)가 칩 선택 신호를 출력한 주변 장치가 동일한 경우 즉, 서브 버스(10)에 접속된 임의의 주변 장치로의 액세스가 연속해서 행해지는 경우 컨트롤러(16)에 후술하는 복원 처리를 시키기 위한 제어 신호를 출력한다.

어드레스 버퍼(21)는 CPU(1)로부터 메인 버스(9)를 거쳐서 공급된 어드레스를 래치한다. 그래서 어드레스 버퍼(21)는 컨트롤러(16)의 제어에 의거해서 래치한 어드레스를 그대로 보존하거나 혹은 소정의 타이밍으로 예를 들면 1개씩 증분 (또는 감산)한다.

다음에 그 동작에 대해서 설명을 하지만, 설명을 알기 쉽도록 하기 위해 실제로는 동시에 행해지는 타이밍의 제어와 버스 폭 및 어드레스 형식의 제어를 나누어서 설명한다.

먼저 타이밍의 제어에 대해서 제 5 도 및 제 6 도의 타이밍 차트를 참조해서 설명을 한다. 또한, 제 5 도 및 제 6 도에 있어서는 신호 Read, DataOk, ChipSelect, SRead, SWrite는 액티브 로우의 논리로 표시하고 있다. 또한, 상지의 각 블록은, 시스템 클록(제 5a 도 및 제 6a 도)의 입상 옛지의 타이밍에 동기해서 동작하는 것으로 한다.

예를 들면 CPU(1)이 디바이스(7)로부터 1 바이트(8 바이트)의 데이터를 판독하는 경우, 제 5 도에 도시하는 바와 같이 CPU(1)로부터 디바이스(7)에 할당된 어드레스(Address)(제 5C 도)가 메인 버스(9)를 거쳐서 출력되고, 이 어드레스는 어드레스 버퍼(21)에 래치됨과 동시에 메인 버스 어드레스 디코더(17)에 입력된다.

어드레스 버퍼(21)에 대치된 어드레스는 신호 SAddress(제 5h 도)로서 서브 버스(10)에 출력됨과 함께 서브 버스 어드레스 디코더(19)로 출력된다.

메인 버스 어드레스 디코더(17) 및 서브 버스 어드레스 디코더(19)에서는 CPU(1)로부터의 어드레스가 디코드되고 그것의 디코드 결과와 어드레스 테이블 기억부(18)에 기억된 어드레스 더코드 테이블이 비교된다.

상술한 바와 같이 이제 CPU(1)로부터는 디바이스(7)에 할당된 어드레스가 출력되고 있으므로 서브 버스 어드레스 디코더(19)에 의해 CPU(1)이 액세스하려고 하는 주변 장치가 디바이스(7)임이 검출되고 서브 버스(10)를 거쳐서 디바이스(7)에 칩 선택 신호(ChipSelect)(제 5f 도)가 출력된다.

서브 버스 어드레스 디코더(19)의 디코드 결과는 상태 버퍼(20)에 기억됨과 동시에 상태 버퍼(20)를 거쳐서 컨트롤러(16)에 출력된다. 이에 따라 컨트롤러 (16)에 의해 CPU(1)의 액세스의 대상이 디바이스(7)임을 인식시킨다.

다시 CPU(1)에서 메인 버스(9)를 거쳐서 기록 신호(Read)(제 5b 도) 및 바이트 액세스를 표시하는 액세스 타입 신호(AccessType)가 컨트롤러(16)에 공급된다.

컨트롤러(16)는 CPU(1)로부터의 판독 신호(Read)를 수신하면 서브 버스(10)에 판독 신호(SRead)(제 5g 도)를 출력한다.

여기에서 제 3 도의 타이밍 테이블에 표시하는 바와 같이 디바이스(7)는 2 클록의 판독 지연 시간을 갖는다. 즉, 디바이스(7)에 있어서는 칩 선택 신호 및 판독 신호(SRead)가 공급되고 데이터를 서브 버스(10)에 출력될 때까지 2 클록만이 걸린다.

거기에서 컨트롤러(16)는 판독 신호(SRead)를 서브 버스(10)로 출력하면 먼저 카운터(15)를 리셋하고 카운터(15)에 시스템 클록(제 5a 도)의 타이밍으로 0으로부터의 카운트를 개시시킨다.

다시 컨트롤러(16)에 있어서 타이밍 테이블 기억부(13)의 타이밍 테이블(제 3 도)에 기술된 디바이스(7)의 판독 지연 시간이 참조되고 카운터(15)의 카운트치가 디바이스(7)의 판독 지연 시간과 같아질 때까지 대기 시간이 놓여진다.

따라서 이 경우 판독 신호(SRead)가 출력되고서부터 2 클록의 대기 시간이 놓여지게 된다.

서브 버스(10)에 판독 신호(SRead)가 출력되고서부터 2 클록 Q분의 시간이 경과하면 디바이스(7)에 있어서 1 바이트의 데이터(SData)(제 5i 도)가 16 비트의 서브 버스(10)의 하위 8 비트에 출력됨과 함께 컨트롤러(16)에 있어서 MUX 12d 및 데이터 버퍼(11)에 서브 버스(10)의 하위 8 비트의 데이터를 래치하도록 CPU(1)로부터의 8 비트(1 바이트)의 액세스 타입을 표시하는 신호 AseessType에 의거한 제어 신호가 출력된다.

이에 따라 디바이스(7)로부터의 1 바이트의 데이터(SData)는 MUX 12d를 거쳐서 데이터 버퍼(11)의 하위 8 비트에 래치된다.

디바이스(7)로부터의 (SData)가 데이터 버퍼(11)에 래치되면 메인 버스(9)에 디바이스(7)로부터의 데이터를 출력할 수가 있음을 표시하는 DataOk 신호(제 5e 도)가 컨트롤러(16)로부터 메인 버스(9)를 거쳐서 CPU(1)에 공급됨과 함께 컨트롤러(16)로부터 서브 버스(10)로 출력되고 있던 판독 신호(SRead)(제 5g 도)가 정지된다.

다시 이때 컨트롤러(16)는 카운터(15)를 리셋하여 이것에 의해 카운터(15)에 있어서 시스템 클록(제 5a도)의 타이밍으로 0으로부터의 카운트가 개시된다.

그래서 데이터 버퍼(11)에 래치된 데이터가 메인 버스(9)위에 출력되고(제 5d 도) 이것이 CPU(1)에 의해 입력이 된다.

그러면 CPU(1)은 메인 버스(9)로의 판독 신호(Read)(제 5b 도) 및 어드레스 (Address)(제 5C 도)의 출력을 정지한다. CPU(1)이 메인 버스(9)로의 판독 신호 (Read)(제 5b) 도의 출력을 정지하면 컨트롤러(16)는 서브 버스 어드레스 디코더 (19)로부터 디바이스(7)로의 칩 선택 신호(ChipSelect)(제 5f 도)의 출력을 정지시킨다.

그래서 컨트롤러(16)는 디바이스(7)로부터의 데이터가 데이터 버퍼(11)에 래치된(데이터의 판독이 종료된) 시각부터 디바이스(7)의 플로팅 시간만이 경과하고 디바이스(7)로부터 서브 버스(10)로의 데이터(SData)(제 5i 도)의 출력이 정지하면 즉 타이밍 테이블(제 3 도)에 기술된 디바이스(7)의 플로팅 타임에 카운터(15)의 카운트치가 같아지면 어드레스 버퍼(21)로부터 서브 버스(10)로의 어드레스 (Address)(제 5h 도)의 출력을 정지시킨다.

그런데 CPU(1)은 상술한 바와 같이 메인 버스(9)로의 어드레스(Address)(제 5C 도)의 출력을 정지한 시점에서 디바이스(7)에 대한 액세스가 종료한 것으로 인식한다(실제로 CPU(1)에 있어서는 종료하고 있다).

따라서, CPU(1)이 메인 버스(9)로의 어드레스(Address)(제 5c 도)의 출력을 정지한 직후 예를 들면 서브 버스(10)에 접속된 메모리(6)에 데이터를 기입하는 경우를 생각하면 디바이스(7)로부터의 데이터가 데이터 버퍼(11)에 래치된(데이터의 판독이 종료된)시각으로부터 디바이스(7)의 플로링 시간 12클록)만이 경과할 때까지는 즉 제 5 도에 있어서 P1로 표시하는 시각을 경과할 때까지는 디바이스(7)가 서브 버스(10) 상에 데이터(SData)를 출력하고 있으므로 이 데이터와 CPU(1)이 메모리(6)로 출력하는 데이터가 서브버스(10) 상에서 배팅하게 된다.

거기에서 이 경우 게이트웨이(4)에 있어서는 CPU(1)이 출력하는 데이터를 데이터 버퍼(11)로 래치하여 두고 메모리(6)로의 기입이 종료한 것을 표시하는 신호 DataOK를 컨트롤러(16)로부터 CPU(1)로 출력하도록 되어 있다.

이에 따라 CPU(1)은 메모리(6)로의 데이터의 기입이 종료하였다고(실제로는 종료하지 아니하였으나)인식하게 된다. 따라서 메인 버스(9)는 즉석에서 개방되고 CPU(1)는 서브 버스(10)에 접속된 주변장치에 액세스라는 처리를 제외하는 처리를 실행할 수가 있도록 된다.

한편, 게이트웨이(4)에 있어서는 디바이스(7)로부터의 데이터가 데이터 버퍼 (11)에 래치된(데이터의 판독이 종료된) 시각부터 디바이스(7)의 플로팅 시간만이 경과하는 것을 기다려(시각 P1 (제 5 도)이 경과하는 것을 기다려), 컨트롤러(16), 서브 버스 어드레스 디코더(19), 어드레스 버퍼(21)로부터 서브 버스(10)를 거쳐서 메모리(6)에 기록 신호(SWrite), 칩 선택신호, 또는 어드레스(SAddress)가 각각 출력된다. 그래서 데이터 버퍼(11)에 래치된 CPU(1)로부터의 데이터가 서브 버스 (10)를 거쳐서 메모리(6)에 출력되어 기입된다.

따라서 이 경우 CPU(1)은 디바이스(7)로부터의 데이터가 데이터 버퍼(11)에 래치된(데이터의 판독이 종료된) 시각부터 디바이스(7)의 플로팅 시간만이 경과하는 것을 기다리지 않고 메모리(6)로의 데이터의 기입 처리를 할 수 가 있으므로 장치전체의 처리속도를 향상시킬 수가 있다.

또 이 경우 메모리(6)로의 데이터의 기입 처리는 말하자면 CPU(1)에서 게이트웨이(4)로 건네져서 게이트웨이(4)에 의해 행해진다. 따라서 CPU(1)은 즉석에서 다른 처리를 실행할 수가 있게되므로 CPU(1)과 주변 장치와의 사이의 액세스 타이밍을 고려할 필요가 없어지고 이에 따라 응용 프로그램의 개발을 용이하게 행할 수가 있도록 된다.

다음으로 제 6도는 CPU(1)로부터 디바이스(8)로의 데이터의 기입 처리를 설명하는 타이밍 차트이다. 먼저 CPU(1)로부터 디바이스(8)에 할당한 어드레스 (Address) (제 6C 도)가 메인 버스(9)를 거쳐서 출력되고 이 어드레스는 어드레스 버퍼(21)에 래치됨과 동시에 메인 버스 어드레스 디코더(17)에 입력된다.

어드레스 버퍼(21)에 래치된 어드레스는 신호 SAddress (제 5h 도)로서 서브 버스(10)로 출력됨과 동시에 서브 버스 어드레스 디코더(19)로 출력된다.

메인 버스 어드레스 디코더(17) 및 서브 버스 어드레스 디코더(19)에서는 CPU(1)로부터의 어드레스가 디코드되고 그것의 디코드 결과의 어드레스 테이블 기억부(18)에 기억된 어드레스 디코드 테이블이 비교된다.

상술한 바와 같이 이제 CPU(1)로부터는 디바이스(8)에 할당된 어드레스가 출력되고 있으므로 서브 버스 어드레스 디코더(19)에 의해 CPU(1) 이 액세스하려고 하는 주변 장치가 디바이스(8)임이 검출되고 서브 버스(10)를 거쳐서 디바이스(8)에 칩 선택신호(ChipSelect) 제6f 도)가 출력된다.

서브 버스 어드레스 디코더(19)의 디코드 결과는 상태버퍼(20)에 기억됨과 함께 상태 버퍼(20)를 거쳐서 컨트롤러(16)에 출력된다. 이에 따라 컨트롤러(16)에 의해 CPU(1)의 액세스의 대상이 디바이스(8)임이 인식된다.

더욱이 CPU(1)에서 메인 버스(9)를 거쳐서 기록 신호(Write) 제(6b 도)가 컨트롤러(16)에 공급됨과 함께 데이터(Data)가 데이터 버퍼(11)에 공급되어서 래치된다.

CPU(1)로부터의 데이터(Data)가 데이터 버퍼(11)에 래치되면 컨트롤러(16)는 CPU(1)에 대해 데이터(Data)의 디바이스(8)로의 기입이 종료한(실제로는 종료하지 않았으나) 것을 표시하는 DataOk 신호(제 6e 도)를 컨트롤러(16)로부터 메인 버스 (9)를 거쳐서 CPU(1)로 공급한다.

CPU(1)은 DataOk 신호를 수신하면 데이터(Data)의 기입이 종료하였다고 인식해서 기록신호(Write), 어드레스(Address) 및 데이터(Data)의 출력을 정지하고 다음의 처리를 실행한다.

한편, 컨트롤러(16)는 CPU(1)로부터의 기록 신호(Write)를 수신하면 서브 버스(10)에 기록신호(SWrite)(제 6g 도)를 출력하는 것과 함께 데이터 버퍼(11)에 래치한 데이터를 신호 SData(제61도)로서 서브 버스(10)로 출력하도록 제어 신호를 출력한다.

이에 따라 데이터 버퍼(11)로부터는 데이터(SData)가 서브 버스(10)위에 출력된다.

여기에서 제 3도의 타이밍 테이블에 도시하는 바와 같이 디바이스(8)는 3 클록의 기록 지연 시간을 갖는다. 즉 디바이스(8)에 있어서는 칩 선택 신호 및 기록신호(SWrite)가 공급되고서부터 데이터를 서브 버스(10)로부터 내장시킬 때까지 3 클록만이 걸린다.

다시 디바이스(8)는 1 클록의 유지 시간을 갖는다. 디바이스(8)에 대해서 데이터의 기입을 행하는 경우에 있어서는 (디바이스(8)에 대한 데이터의 기입을 보상하는데는), 기록 신호(SWrite)의 공급을 정지하고부터 최소한 1 클록 분의 시간이 경과할 때까지는 데이터(SData) 및 어드레스(SAddress)를 서브 버스(10)위에 계속해서 공급할 필요가 있다.

거기에서 컨트롤러(16)는 기록 신호(SWrite) (제 6g 도)를 서브 버스(10)에 출력하면 먼저 카운터(15)를 리셋하여 카운터(15)에 시스템 클록(제 6a 도)의 타이밍으로 0으로부터의 카운트를 개시시킨다.

다시 컨트롤러(16)에 있어서 타이밍 테이블 기억부(13)의 타이밍 테이블(제 3 도)에 기술된 디바이스(8)의 기록지연 시간이 참조되고 카운터(15)의 카운트치가 디바이스(8)의 기록 지연 시간과 같게 될 때까지 대기 시간이 넣어진다.

따라서 이 경우 기록 신호(SWrite)가 출력되어서부터 3 클록의 대기 시간이 놓여지게 된다.

서브 버스(10)로 기록 신호(SWrite)가 출력되고서부터 3 클록분의 시간이 경과하면 컨트롤러(16)로부터의 기록 신호(SWrite) 및 칩 선택 신호의 출력이 정지된다.

컨트롤러(16)는 기록 신호(SWrite) (제 6g 도)의 출력을 정지하면 카운터 (15)를 리셋하고 카운터(15)에 시스템클록(제 6a 도)의 타이밍으로 0으로부터의 카운트를 개시시킨다.

그래서 컨트롤러(16)에 있어서 타이밍 테이블 기억부(13)의 타이밍 테이블 (제 3 도)에 기술된 디바이스(8)의 유지 시간이 참조되고 카운터(15)의 카운트치가 디바이스(8)의 유지시간과 같아질 때까지 대기 시간이 놓여진다.

따라서 이 경우 기록 신호(SWrite)가 출력이 정지되고서부터 1 클록의 대기시간이 놓여지게 된다.

기록 신호(SWrite)에 출력이 정지되고서부터 1 클록분의 시간이 경과하면(1 클록분 이상의 시간이 경과한 후의 시스템 클록의 최초의 입상 엣지의 타이밍(제 6 도에 있어서, P2로 표시하는 시각)으로 데이터 버퍼(11)또는 어드레스 버퍼(21) 각각으로부터의 데이터(SData) 또는 어드레스(SAddress)의 출력이 정지되고 이에 따라 디바이스(8)로의 데이터의 기입처리가 종료한다.

이상과 같이해서 유지시간을 갖는 디바이스(8)에 대한 액세스 타이밍이 제어되고 디바이스(8)에 대한 데이터의 기입이 보증된다(디바이스(8)에 데이터가 정확하게 기입된다).

다시 디바이스(8)로의 데이터의 기입 처리는 말하자면 CPU(1)로부터 게이트웨이(4)에 넘겨지고 게이트웨이(4)에 의해 행해진다. 따라서 상술한 바와 같이 CPU(1)은 즉시로 다른 처리를 실행할 수가 있게 되므로 장치전체의 처리 속도를 향상시킬 수가 있다.

다음으로 예를 들면 메모리(6)에 2회 연속해서 데이터의 기입이 행해지는 경우(2회 연속해서 메모리(6)로의 액세스가 행해지는 경우)먼저 1회째는 상술하는 디바이스(8)로의 데이터의 기입 처리에 있어서의 경우와 같은 처리가 행해진다.

즉 게이트웨이(4)에 있어서 메모리(6)의 기록 지연시간(제 3 도)을 고려한 타이밍 제어가 행해진다(메모리(6)의 유지시간(제 3도)은 0 이므로 무시된다).

여기에서 제 3 도의 타이밍 테이블에 도시한 바와 같이 메모리(6)는 10 클록의 복원 시간을 갖는다. 즉, 메모리(6)에 2회 연속해서 액세스하는 경우에 있어서는 1 회 째의 액세스가 종료하고서부터 10 클록 분의 시간을 경과할 때까지는 2회 째의 액세스가 보충되지 않는다.

거기에서 게이트웨이(4)에 있어서는 메모리(6)로의 1회 째의 액세스의 종료 후 메모리(6)로의 2 회 째의 액세스가 개시되고 상태 버퍼(20)에 서브 버스 어드레스 디코더(19)가 칩 선택 신호를 출력한 주변 장치에 할당된 어드레스의 디코드 결과가 공급되면 그것의 리코드 결과와 지금 상태버퍼(20)에 기억되어 있는 기억치가 비교된다.

여기에서 1회 째의 메모리(6)로의 데이터의 기입 처리가 종료하였을 때에는 상태버퍼(20)에 서브 버스 어드레스 디코더(19)의 최신의 디코드 결과 즉 바로 앞에 액세스된 주변 장치가 메모리(6)임이 기억되어 있다.

따라서 이 경우 서브 버스 어드레스 디코더(19)로부터 출력된 디코드 결과와 상태 버퍼(20)에 지금 기억되어 있는 기억치가 동일한 것으로 판정된다.

그러면 컨트롤러(16)는 복원 처리를 한다.

즉 컨트롤러(16)는 먼저 카운터(16)를 리셋하여 카운터(15)에 시스템 클록의 타이밍으로 0으로부터의 카운트를 개시시킨다.

동시에 컨트롤러(16)에 있어서 타이밍 테이블 기억부(13)의 타이밍 테이블 (제 3 도)에 기술된 메모리(6)의 복원시간이 참조되고 그것이 0 보다 큰 값으로 설정되어 있으면 카운터(15)의 카운트치가 메모리(6)의 복원 시간과 같아질 때까지 대기 시간에 놓여진다.

따라서 이 경우 메모리(6)로의 1회 째의 액세스가 종료하고서부터 10 클록 분의 대기 시간(간격)이 놓여지고 그 후에 메모리(6)에 대한 2 회 째의 데이터의 기입이 행해진다.

이상과 같이하여 복원 시간을 갖는 메모리(6)에 대한 액세스 타이밍이 제어되고 메모리(6)에 대해서 연속해서 액세스가 행해져도 그것이 동작이 보증된다.

또한 게이트웨이(4)에 의한 메모리(6)로의 1 회 째의 액세스의 종료 후 CPU(1) 이 메모리(6)에 대해서 2 회 째의 기입 데이터로서 출력하는 데이터는 게이트웨이(4)의 데이터 버퍼(4)로 곧바로 래치된다.

그러면 컨트롤러(16)로부터는 신호 DataOk가 CPU(1)로 출력된다.

이에 따라 CPU(1) 는 메모리(6)로의 데이터의 기입이 종료한(실제로는 데이터 버퍼(11)에 래치된 것만으로 메모리(6)로의 데이터의 기입은 종료되지 않음)것으로 인식하고 메인 버스(9)는 개방된다. 따라서 CPU(1) 은 게이트웨이(4)에 의한 메모리(6)로의 1 회 째의 액세스의 종료 후 곧바로 다른 처리를 실행할 수가 있으므로 장치 전체의 처리 속도를 향상시킬 수가 있다.

또한 타이밍 테이블 기억부(13)에 기억된 타이밍 테이블(제 3 도)은 소정의 프로그램을 CPU(1) 에 실행시킴으로써 고쳐 쓸 수가 있도록 되어있다. 따라서 예를 들면 ROM(5), 백업 메모리(6), 디바이스(7, 8)의 사양이 변경되거나 새로운 저변 장치가 추가되어도 타이밍 테이블을 고쳐 쓰는 것만으로 끝나고 예를 들면 이제까지 사용하고 있던 용융 소프트의 호환성이 없어지고 사용 불가능해지는 일이 방지된다.

다음에 버스 폭 및 어드레스 형식의 제어에 대해서 제 7 도 및 제 8 도의 타이밍 차트를 참조해서 설명한다. 또한, 제 7 도 및 제 8 도에 있어서는 신호 Read, DataOk, ChipSelect, SRead, SWrite 는, 액티브로우의 논리로 표시하고 있다. 또한 장치의 각 블록은 시스템 클록(제 7a 도 및 제 8a도)의 입상 엣지의 타이밍에 동기해서 동작하는 것으로 한다.

예를 들면 CPU(1) 가 8 비트의 ROM(5)로부터 32 비트의 데이터를 판독하는 경우 제 7 도에 도시하는 바와 같이 CPU(1)로부터 ROM(5)로부터 판독하려는 데이터의 어드레스(Address)(제 7C 도)가 메인 버스(9)를 거쳐서 출력되고, 이 어드레스는 어드레스 버퍼(21)에 래치됨과 동시에 메인 버스 어드레스 디코더(17)에 입력된다.

어드레스 버퍼(21)에 래치된 어드레스는 신호 SAddress(제 7h 도)로서 서브 버스(10)에 출력됨과 함께 서브 버스 어드레스 디코더(19)에 출력된다.

메인 버스 어드레스 디코더(17) 및 서브 버스 어드레스 디코더(19)에서는 CPU(1)로부터의 어드레스가 디코드되고 그것의 디코드 결과와 어드레스 테이블 기억부(18)에 기억된 어드레스 디코드 테이블이 비교된다.

상술한 바와 같이 CPU(1)로부터는 ROM(5)의 어드레스가 출력되어 있으므로 서브 버스 어드레스 디코더(19)에 의해 CPU(1) 이 액세스하려고 하는 주변 장치가 ROM(5) 임이 검출되고 서브 버스(10)를 거쳐서 ROM(5)에 칩 선택 신호 (ChipSelect)(제 7f 도)가 출력된다.

서브 버스 어드레스 디코더(19)의 디코드 결과는 상태 버퍼(20)에 기억됨과 동시에 상태 버퍼(20)를 컨트롤러(16)로 출력된다. 이에 따라 컨트롤러(16)에 의해 CPU(1) 의 액세스의 대상이 ROM(5)인 것으로 인식된다.

다시 CPU(1)로부터 메인 버스(9)를 거쳐서 판독 신호(Read)(제 7b 도) 및 판독하는 데이터 길이가 32 비트임을 표시하는 액세스 타입 신호(AccessType)가 컨트롤러(16)에 공급된다.

컨트롤러(16)는 CPU(1)로부터의 판독 신호(Read)를 수신하면 서브 버스(10)에 판독 신호(SRead)(제 7g 도)를 출력한다.

이에 따라 ROM(5) 의 어드레스(SAddress)로부터 8 비트의 데이터(SData)(제 7i 도)가 16 비트의 서브 버스(10)의 하위 8 비트로 출력된다.

여기에서 제 4 도의 버스 폭 테이블에 도시하는 바와 같이 ROM(5) 는 그것의 버스 폭이 8 비트이고 어드레스 형식이 가변으로 되어있다. 즉, ROM(5) 로부터 액세스 타입 신호(AccessType)로 표시되는 32 비트의 데이터를 판독하려면 버스 폭 단위로서의 8 비트 단위로 어드레스를 증가(또는 감소)시키면서 4 회 판독하는 처리를 할 필요가 있다.

거기에서 컨트롤러(16)는 CPU(1) 로부터의 액세스 타입 신호(AccessType) 및 버스 폭 테이블 기억부(14)의 버스 폭 테이블(제 4 도)에 기술된 ROM(5) 의 버스 폭 및 어드레스 형식에 의거해서 다음과 같은 제어를 한다.

즉, 32 비트의 데이터가 4 개의 8 비트로 분할된 데이터를 상위의 8 비트로부터 d_4, d_3, d_2, d_1로 표시한 경우 이 4 개의 8 비트의 데이터 d_4, d_3, d_2, d_1가 ROM(5) 의 어드레스(SAddress+3), (SAddress+2), (SAddress+1), (SAddress)에 각각기입되어 있다면 컨트롤러(16)는 먼저 상술하는 바와 같이해서 서브 버스(10)에 출력된 ROM(5)의 어드레스(SAddress)로부터의 8 비트의 데이터를 MUX12d를 거쳐서 32 비트의 데이터 버퍼(11)의 하위 8 비트에 래치시킨다.

그래서 컨트롤러(16)는 어드레스 버퍼(21)에 래치되어 있는 어드레스 (SAddress (=Address)를 1 만큼 증분시켜 어드레스(SAddress+1)로서 서브 버스(10)를 거쳐서 ROM(5) 로 출력시킨다.

이에 따라 ROM(5)의 어드레스(SAddress+1)로부터 8 비트의 데이터가 서브 버스(10)에 출력된다. 이 8 비트의 데이터는 MUX(12c)를 거쳐서 데이터 버퍼(11)에 그것의 하위 8 비트의 다음 8 비트의 데이터로서 래치된다.

이하, 동일한 방식으로, ROM(5)의 어드레스(SAddress+2), (SAddress+3)로부터 8 비트의 데이터가 판독되고 데이터 버퍼(11)에 차례로 래치된다. 또한 이때 컨트롤러(16)는 어드레스 버퍼(21)에 기억된 어드레스를 증분하는 타이밍에 의거해서 제 7 g 도에 도시하는 바와 같이 ROM(5) 에 대한 판독 신호(SRead)를 ON/OFF 한다.

이상과 같이해서 ROM(5) 의 어드레스(SAddress), (SAddress+l), (SAddress+2), (SAddress+3)로부터 각각 판독된 8 비트의 데이터가 데이터 버퍼 (11)에 래치되고 32 비트의 데이터로 되면, 즉 4 개의 8 비트의 데이터가 32 비트의 데이터에 팩되면 컨트롤러(16)는 메인 버스(9)를 거쳐서 CPU(1) 에 신호 DataOk(제 7e 도)를 출력하고 다시 데이터 버퍼(11)에 래치되어 있는 32 비트의 데이터(제 7d 도)를 메인 버스(9)에 출력시킨다.

CPU(1)은 신호 DataOk를 수신하면 메인 버스(9)위의 32 비트의 데이터를 내장시켜 이에 따라 ROM(5)로부터의 32 비트의 데이터의 입력이 종료한다.

또한, 버스 폭 테이블 기억부(14)에 기억된 버스 폭 테이블(제 4 도)은 소정의 프로그램을 CPU(1)에 실행시킴으로써 고쳐 쓸 수가 있도록 되어있다. 따라서 예를 들면 ROM(5)가 8 비트의 ROM으로부터 16 비트의 것으로의 변경된 경우에는 버스 폭 테이블(제 4 도)의 ROM(5)의 버스 폭을 8 비트로부터 16 비트로 바꾸는 것만으로 좋다.

이 경우 ROM(5)로부터 2 개의 16 비트의 데이터가 판독되고 그 중의 한 편 (예를 들면 어드레스가 하위의 16 비트의 데이터)은 MUX(12c 및 12d)를 거쳐서 데이터 버퍼(11)의 하위 16 비트에 래치되고 그것의 다른 편(예를 들면 어드레스가 상위의 16 비트의 데이터)은 MUX12a 및 12b를 거쳐서 데이터 버퍼(11)의 상위 16 비트에 래치되어 32 비트에 팩된다.

다음으로 CPU(1)에 의해 디바이스(7)로의 32 비트의 데이터의 기입이 행해지는 경우 제 8 도에 도시하는 바와 같이 CPU(1)에 의해, 디바이스(7)로 데이터를 기입하려고 하는 어드레스(address) (제 8c 도)가 메인 버스(9)를 거쳐서 출력되고, 이 어드레스는 버퍼(21)에 래치됨과 동시에 메인 버스 어드레스 디코더(17)에 입력된다.

어드레스 버퍼(21)에 래치된 어드레스는 신호 SAddress(제 8h 도)로서 서브 버스(10)에 출력됨과 동시에 서브 버스 어드레스 디코더(19)에 출력된다.

메인 버스 어드레스 디코더(17) 및 서브 버스 어드레스 디코더(19)에서는 CPU(1)로부터의 어드레스가 디코드되고 그것의 디코드 결과와 어드레스 테이블 기억부(18)에 기억된 어드레스 디코드 테이블이 비교된다.

상술한 바와 같이 CPU(1)로부터 디바이스(7)의 어드레스가 출력되고 있으므로 서브 버스 어드레스 디코더(19)에 의해 CPU(1) 이 액세스하려고 하는 주변 장치가 디바이스(7)임이 검출되고 서브 버스(10)를 거쳐서 디바이스(7)에 칩 선택 신호 (ChipSelect) (제 8f 도)가 출력된다.

서브 버스 어드레스 디코더(19)의 디코드 결과는 상태 버퍼(20)에 기억됨과 함께 상태 버퍼(20)를 거쳐서 컨트롤러(16)에 출력된다. 이에 따라 컨트롤러(16)에 의해 CPU(1) 의 액세스의 대상이 디바이스(7)임이 인식된다.

다시 CPU(1)로부터 메인 버스(9)를 거쳐서 기록 신호(Write) (제 8b 도) 및 기입 데이터 길이가 32 비트임을 표시하는 액세스 타입 신호(AccessType)가 컨트롤러(16)에 공급됨과 동시에 디바이스(7)에 기입되는 32 비트의 데이터(Data) (제 8d 도)가 메인 버스(9)에 출력된다.

컨트롤러(16)는 CPU(1)로부터의 기록 신호(Write)를 수신하면 메인 버스(9)상의 32 비트의 데이터를 데이터 버퍼(11)에 래치시킴과 동시에 서브 버스(10)에 기록 신호(SWrite) (제 8g도)를 출력한다.

다시, 컨트롤러(16)는 CPU(1)에 신호 DataOk를 출력한다. 이에 따라 CPU(1)은 디바이스(7)로의 데이터의 기입이 종료하였다고(실제로는 종료하지 않았으나) 인식하고 메인 버스(9)를 개방된다. 따라서 그 후 CPU(1)는 그 밖의 처리를 곧바로 실행할 수가 있으므로 장치 전체의 처리속도를 향상시킬 수가 있다.

여기에서, 제 4 도의 버스 폭 테이블에 도시하는 바와 같이 디바이스(7)는 그것의 버스 폭이 8 비트이고, 어드레스 형식이 고정되어 있다. 즉 디바이스(7)로, 액세스 타입 신호(AccessType)로 표시되는 32 비트의 데이터를 기입하는 데에는, 이것을 디바이스(7)의 버스 폭 단위로서의 8 비트의 데이터로 4 분할하고 이 4 개의 8 비트의 데이터를 어드레스를 고정해서 기입하도록 할 필요가 있다.

거기에서 컨트롤러(16)는 CPU(1)로부터의 액세스 타입 신호(AccessTyPe)와 버스 폭 테이블 기억부(14)의 버스 폭 테이블(제 4 도)에 기술된 디바이스(7)의 버스 폭 및 어드레스 형식에 의거해서 다음과 같은 제어를 한다.

즉 컨트롤러(16)는 데이터 버퍼(11)에 래치되어 있는 32 비트의 데이터를 8 비트 폭의 MUX12a 내지 12d 에 각각 판독시킴으로써 4 개의 8 비트의 데이터로 언팩한다.

그래서 컨트롤러(16)는 먼저 MUX(12d)로 판독시킨 32 비트의 데이터 중 LBS 측의 8 비트의 데이터를 서브 버스(10)의 하위 8 비트로 출력시킨다, 이에 따라 디바이스(7)의 어드레스(SAddress)에 8 비트의 데이터가 기록된다.

다시 컨트롤러(16)는 어드레스 버퍼(21)에 기억하고 있는 어드레스 (SAddress)를 유지시킨 채, MUX(12c)에 판독시켜, 32 비트의 데이터 중의 LSB 측의 8 비트 다음의 상위 8 비트의 데이터를 서브 버스(10)의 하위 8 비트로 출력시킨다. 이에 따라 디바이스(7)의 어드레스(SAddress)에, 8 비트의 데이터가 기록된다.

이하, 동일한 방식으로, 디바이스(7)의 어드레스(SAddress)에 MUX(12b 및 12a)에 데이터 버퍼(11)로부터 판독시킨 8 비트의 데이터가 차례로 기록된다. 또한, 이때 컨트롤러(16)는 MUX(12a 내지 12d)에 데이터 버퍼(11)로부터 데이터를 판독시키는 타이밍에 의거해서 제 8g 도에 도시하는 바와 같이 디바이스(7)에 대한 기록 신호(SWrite)를 ON/OFF 한다.

디바이스(7)는 예를 들면 FIFO 형의 버퍼를 내장하고 있고 이 버퍼에 어드레스(SAddress)에 대해서 기입된 4 개의 8 비트의 데이터가 차례로 스택되고(쌓여 넣어지고) 그 후 디바이스(7)에 있어서 소정의 처리가 실시된다.

또한, 예를 들면 디바이스(7)가 버스 폭 8 비트의 것으로부터 16 비트의 것으로 변경된 경우에는 상술한 경우와 같이 해서 버스 폭 테이블(제 4 도)의 디바이스(7)의 버스 폭을 8 비트에서 16 비트로 바꾸는 것만으로 좋다.

이 경우 먼저 데이터 버퍼(11)에 래치된 32 비트의 데이터 중 하위 16 비트의 데이터가 MUX(12c 및 12d)에 의해 판독되고, 다시 그것의 상위 16 비트의 데이터가 MUX12a 및 12b 에 의해 판독됨으로써 32 비트의 데이터가 2 개의 16 비트의 데이터에 언팩된다.

따라서 주변 장치의 버스 폭 또는 어드레스 방식이 변경되거나 새로운 주변 장치가 추가되어도 버스 폭 테이블을 고쳐 쓰는 것만으로 끝나고, 예를 들면 이제까지 사용하고 있던 응용 소프트웨어의 호환성이 없어지고 사용 불가능으로 되는 것이 방지된다.

이상 설명을 알기 쉽게 하기 위해서 타이핑의 제어와 버스 폭 및 어드레스 형식의 제어를 나누어서 설명을 하였으나 게이트웨이(4)에서는 타이밍 테이블(제 3 도) 및 버스 폭 테이블(제 4 도)에 의거해서 제 9 도의 플로차트에 도시하는 바와같이 이들의 제어가 병렬로서(동시에) 행해지도록 되어있다.

즉 게이트웨이(4)에 있어서는 먼저 단계 S1 에 있어서 CPU(1)로부터 메인 버스(9)를 거쳐서 기록 신호 또는 판독 신호와 함께 어드레스가 출력되는 것이 기대되고 CPU(1)로부터 기록 신호 또는 판독 신호와 함께 어드레스가 출력되면 단계 S2로 진행하여 주변 장치에 대해서 CPU(1)로부터의 어드레스가 출력됨과 동시에 그것의 어드레스의 디코드가 행해지고 그것의 디코드 결과에 대응하는 주번 장치에 칩 선택 신호가 출력된다.

그래서 단계 S3으로 진행하고 CPU(1)로부터의 어드레스를 디코드한 결과로부터 CPU(1) 이 액세스하려고 하고 있는 주변 장치가 서브 버스(10)에 접속되었는지의 여부가 판별된다.

단계 S3에 있어서 CPU(1)이 액세스하려고 하는 주변 장치가 서브 버스(10)에 접속된 것이 아니라고 판정된 경우 즉 CPU(1)이 액세스하려고 하는 주변 장치가 메인 버스(9)에 접속된 것인 경우 상술한 복원 처리를 할 필요가 없기 때문에 단계 S4로 진행하고 상태 버퍼(20)가 소거되고 단계 S5로 진행한다.

단계 S5에 있어서 메인 버스(9)에 접속된 주변 장치에 대한 CPU(1)의 액세스가 행해지고 그것이 종료하면 단계 S1로 되돌아가 다시 단계 S1로부터의 처리를 되풀이한다.

또한 단계 S3 에 있어서 CPU(1)가 액세스하려 하고 있는 주변 장치가 서브 버스(10)에 접속된 것으로 판정된 경우 단계 S6, S7로 차례로 진행하고 타이밍 테이블 기억부(13), 버스 폭 테이블 기억부(14)로부터 타이밍 테이블 버스 폭 테이블이 각각 판독된다.

그래서 단계 S8로 진행하고 CPU(1)의 액세스 타입과 CPU(1)가 액세스하려 하는 주변 장치의 버스 폭이 같은가 다른가가 판정된다. 단계 S8에 있어서 CPU(1) 의 액세스 타입과 CPU(1)이 액세스하려 하는 주변 장치의 버스 폭이 같다고 판정된 경우 단계 S10으로 진행하고 변수 N 에 1 이 세트되고 단계 S11로 진행한다.

또한 단계 S8 에 있어서 CPU(1)의 액세스 타입과 CPU(1)으로 액세스하려 하는 주변 장치의 버스 폭이 같지 않다고 판정된 경우 단계 S9로 진행하고 변수 N에 팩/언팩을 하는 횟수가 세트되고, 단계 S11로 진행한다.

단계 S11에 있어서는 단계(1)에서 CPU(1)로부터 출력된 신호가 기록 신호인가 또는 판독 신호인가가 판정된다.

단계 S11에 있어서 단계(1)에서 CPU(1)로부터 출력된 신호가 판독 신호라고 판정이 된 경우 단계 S13으로 진행하고 주변 장치로부터의 데이터의 판독 처리가 행해진다.

즉, 단계 S13 에서는 먼저 제 10 도에 도시하는 단계 S21 에 있어서 복원 처리가 행해진다.

여기에서 단계 S13 의 복원 처리는 제 11 도의 플로 차트에 도시하는 바와 같이 행해진다.

즉, 먼저 단계 S31에 있어서, 상태 버퍼(21)가 참조되고 이제 CPU(1)이 액세스하려 하는 주변 장치가 그 직전에 CPU(1)에 액세스한 주변 장치인지의 여부가 판정된다.

단계 S31에 있어서 이제 CPU(1) 이 액세스하려 하는 주변 장치가 그 직전에 CPU(1) 이 액세스한 주변 장치가 아니라고 판정이 된 경우 단계 S32로 진행하고 상태 버퍼(20)에 지금 CPU(1) 이 액세스 하려하고 있는 주변 장치가 기억되고 처리를 종료한다.

또 단계 S31에 있어서 이제 CPU(1)가 액세스하려 하는 주변 장치가 그것의 바로 앞에 CPU(1)가 액세스한 주변 장치라고 판정이 된 경우 단계 S33으로 진행하고 타이밍 테이블(제 3 도)이 참조되고 이제 CPU(1)가 액세스하려 하는 주변 장치가 복원 시간을 갖는지의 여부가 판정된다.

단계 S33에 있어서 이제 CPU(1)가 액세스하려 하는 주변 장치가 복원 시간을 갖지 않았다고 판정이 된 경우, 처리를 종료한다.

단계 S33에 있어서 이제 CPU(1)가 액세스하려하고 있는 주변 장치가 복원 시간을 갖는다고 판정된 경우 단계 S34로 진행하고 그것의 복원 시간분의 대기 시간이 놓여지고 처리를 종료한다.

이상과 같이하여 단계 S31 내지 S34 의 복원 처리가 종료하면, 즉 제 10 도의 단계 S21의 처리가 종료하면 단계 S22로 진행하고 CPU(1)가, 데이터를 판독하려하는 주변 장치에 서브 버스(10)를 거쳐서 리드 신호(SRead)가 출력되고 단계S23으로 진행한다.

단계 S23에 있어서 타이밍 테이블 기억부(13)의 타이밍 테이블에 기술되고, CPU(1)가 데이터를 판독하려 하는 주변 장치의 판독 지연 시간분의 대기 시간이 놓여지고, 단계 S24로 진행하여 주변 장치로부터 데이터가 판독되고, 그것이 데이터버퍼(11)에 그것의 하위 비트 편으로부터 래치됨과 동시에 판독 신호(SRead)의 출력이 정지된다.

그래서 단계 S25로 진행하고 이제 데이터의 판독을 한 주변 장치가 플로팅 타임을 갖는지의 여부가 판정된다. 단계 S25에 있어서 이제 데이터의 판독을 행한 주변 장치가 플로팅 타임을 가졌다고 판정된 경우 단계 S26으로 진행하고 그것의 플로팅 타임 분의 대기 시간이 놓여지고 단계 S27로 진행한다.

단계 S25 에 있어서 이제 데이터의 판독을 행한 주변 장치가 플로팅 타임을 갖지 않았다고 판정이 된 경우 단계 S26을 스킵해서 단계 S27로 진행하고 변수 N 이 1 만큼 감소된다.

그래서 단계 S28로 진행하고 변수 N 이 0 과 같은 지의 여부가 판정된다. 단계 S28 에 있어서 변수 N 이 0 과 같지 않다고 판정된 경우 단계 S29로 진행하고 주변 장치로부터 데이터를 다시 판독하기 위한 다음의 어드레스의 출력 처리가 행해진다.

즉, 단계 S29 에서는 먼저 제 12 도에 도시하는 단계 S41에 있어서 제 9 도의 단계 S7에서 버스 폭 테이블 기억부(14)로부터 판독한 버스 폭 테이블(제 4 도)이 참조되고, 이제 데이터의 판독을 행한 주변 장치의 어드레스 형식이 고정인지의 여부가 판정된다.

단계 S41에 있어서 이제 데이터 판독을 행한 주변 장치의 어드레스 형식이 고정이라고 판정된 경우, 즉 그것의 주변 장치에서 데이터를 다시 읽어내는 데이터 어드레스 버퍼(21)에 기억된 어드레스를 증감시킬 필요가 없는 경우 단계 S42를 스킵해서 처리를 종료한다.

또한 단계 S41에 있어서 이제 데이터의 판독을 행한 주변 장치의 어드레스 형식이 고정이 아니고 가변이라고 판정된 경우, 단계 S42로 진행하고 어드레스 버퍼(21)에 기억된 어드레스가 1 만큼 증분되고 처리를 종료한다.

이상과 같이하여 다음의 어드레스의 출력 처리가 종료한 후, 즉 제 10 도의 단계 S29의 처리를 종료한 후, 단계 S21로 되돌아가 단계 S28에 있어서 변수 N 이 0 과 같다고 판정될 때까지 단계 S21 내지 S29 의 처리를 되풀이한다.

즉, 단계 S21 내지 S29 에 있어서 주변 장치로부터 그것의 버스 폭의 데이터가 차례로 판독되어 그것의 데이터가 데이터 버퍼(11)에 그것의 하위 비트의 편에서 차례로 래치됨으로써 팩된다. 그래서 이 처리가

(CPU(1) 의 액세스 타입)/(주변 장치의 버스 폭)회

만큼 반복한다.

한편 단계 S28에 있어서 변수 N 이 0 과 같다고 판정이 된 경우, 즉 단계 S21 내지 S29 의 처리에 의해 데이터 버퍼(11)에 그것의 하위 비트로부터 차례로 래치된 데이터의 데이터 길이가 CPU(1)의 액세스 타입과 같아진 경우 단계 S30으로 진행하고 데이터 버퍼(11)에 래치되어 있는 데이터가 메인 버스(9)를 거쳐서 CPU(1) 로 공급되고 처리를 종료한다.

이상과 같이해서 주변 장치로부터의 데이터의 판독 처리가 종료하면 즉 제 9 도의 단계 S13 의 처리가 종료하면 단계 S14로 진행한다.

한편 제 9 도의 단계 S11 에 있어서 단계 1 에서 CPU(1) 로부터 출력된 신호가 기록 신호라고 판정된 경우 단계 S12로 진행하고 주변 장치로의 데이터의 기록 처리가 행해진다.

즉, 단계 S12 에서는 먼저 CPU(1)로부터 메인 버스(9)를 거쳐서 출력된 데이터가 데이터 버퍼(11)에 래치된다. 그래서 제 13 도에 도시하는 단계 S51 에 있어서 CPU(1) 에 대해 신호 DataOk 가 출력되고 단계 S52 로 진행한다.

이에 따라 상술한 바와 같이 CPU(1)은 주변 장치로의 데이터의 기입이 종료하였다고(실제로는 종료하지 않았으나) 인식한다.

단계 S52에 있어서 상술한 제 10 도의 단계 S21(제 11 도의 단계 S31 내지 S34)과 같은 복원 처리가 행해지고 단계 S53으로 진행하고, 데이터 버퍼(11)에 래치된 CPU(1)로부터의 데이터가 이제 그것의 데이터의 기입을 하는 주변 장치의 버스 폭(본 실시예에서는 8 비트 또는 16 비트)에 맞추어서 그것의 하위 비트 편에서 MUX(12a 내지 12d)를 거쳐서 판독되고 신호 SData로서 서브버스(10)로 출력됨과 동시에 기록 신호(SWrite)가 서브 버스(10)에 출력된다.

그래서 단계 S54로 진행하고 제 9 도의 단계 S6에서 타이밍 테이블 기억부 (13)에서 판독된 타이밍 테이블에 기술된 지금 데이터의 기입을 하려하고 있는 주변 장치의 기록 지연 시간이 참조되고 그것의 분만큼의 대기 시간이 놓여진다.

이에 따라 주변 장치의 버스 폭에 맞추어서 데이터 버퍼(11)로부터 판독된 데이터가 그것의 주변 장치로 기입된다.

그 후 단계 S55에 있어서 기록 신호(SWrite)의 출력이 정지되고 단계 S56으로 진행하고 이제 데이터의 기입을 한 주변 장치가 유지 시간을 갖는지의 여부가 판정된다. 단계 S56에 있어서, 이제 데이터의 기입을 한 주변 장치가 유지 시간을 갖는다고 판정된 경우 단계 S57로 진행하고 그것의 유지 시간만큼의 대기 시간이 놓여지고 단계 S58로 진행한다.

한편 단계 S56 에 있어서 지금 데이터의 기입을 한 주변 장치가 유지 시간을 갖지 않는다고 판정된 경우 단계 S57을 스킵해서 단계 S58로 진행하고 변수 N 이 1 만큼 감소된다.

그래서 단계 S59로 진행하고 변수 N 이 0 과 같은가의 여부가 판정된다. 단계 S59 에 있어서 변수 N 이 0 과 같지 않다고 판정된 경우 단계 S60으로 진행하고 주변 장치에 데이터를 다시 써넣기 위한 다음의 어드레스, 데이터의 출력 처리가 행해진다.

즉, 단계 S60에서는 먼저 제 14 도에 도시하는 단계 S71 에 있어서 제 9 도의 단계 S7에서 버스 폭 테이블 기억부(14)에서 판독된 버스 폭 테이블(제 4 도)이 참조되고 이제 데이터의 기입을 한 주변 장치의 버스 폭이 8 비트인지의 여부가 판정된다.

단계 S71에 있어서 이제 데이터의 기입을 한 주변 장치의 버스 폭이 8 비트가 아니라고 판정된 경우, 즉 그것의 버스 폭이 16 비트인 경우(본 실시예에 있어서는 서브 버스(10)에 접속된 주변 장치의 버스 폭이 8 비트 또는 16 비트의 어떤것으로 하고 있음), 단계 S72로 진행하여 데이터 버퍼(11)에 래치된 32 비트의 데이터의 상위 16 비트가 MUX(12a, 12b)로 출력되고 단계 S74로 진행한다.

또한, 단계 S71에 있어서, 이제 데이터의 기입을 한 주변 장치의 버스 폭이 8 비트라고 판정된 경우 단계 S73으로 진행하고 데이터 버퍼(11)에 래치된 32 비트의 데이터중의 바로 앞에 출력된 8 비트의 다음의 상위 8 비트가 MUX12c(12b 또는 12a)로 출력되고 단계 S74로 진행한다.

단계 S74에 있어서는 다시 버스 폭 테이블(제 4 도)이 참조되고, 이제 데이터의 기입을 한 주변 장치의 어드레스 형식이 고정인지의 여부가 판정된다.

단계 S74에 있어서 이제 데이터의 기입을 한 주변 장치의 어드레스 형식이 고정이라고 판정이 된 경우 즉 그것의 주변 장치에 데이터를 다시 기입하는 데에 어드레스 버퍼(21)에 기억된 어드레스를 증감시킬 필요가 없는 경우 단계 S75를 스킵해서 처리를 종료한다.

또한 단계 S74에 있어서 이제 데이터의 기입을 한 주변 장치의 어드레스 형식이 고정이 아니고 가변이라고 판정된 경우 단계 S75로 진행하고 어드레스 버퍼 (21)에 기억된 어드레스가 1 만큼 증분되고 처리를 종료한다.

이상과 같이 하여 다음의 어드레스, 데이터의 출력 처리가 종료한 후 즉 제 13 도의 단계 S60의 처리가 종료한 후 단계 S51로 되돌아가 단계 S59에 있어서 변수 N이 0과 같다고 판정이 될 때까지 단계 S51 내지 S60의 처리를 되풀이한다.

즉, 단계 S51 내지 S60에 있어서 CPU(1)로부터의 데이터를 기억하고 있는 데이터 버퍼(11)로부터 주변 장치의 버스 폭에 맞추어서 그것의 하위 비트 편으로부터 데이터가 차례로 판독됨으로써 데이터가 언팩되고 주변 장치에 기입이 된다. 그래서 이 처리가,

(CPU(1)의 액세스 타입)/(주변 장치의 버스 폭)회

만큼 되풀이된다.

한편, 단계 S59에 있어서 변수 N이 0과 같다고 판정된 경우 즉 단계 S51 내지 S60의 처리에 의해 데이터 버퍼(11)의 하위 비트로부터 데이터가 차례로 판독되고 그것의 데이터가 주변 장치에 모두 기입된 경우 단계 S61로 진행하고 주변 장치로의 데이터(SData)의 출력이 정지되어 처리를 종료한다.

이상과 같이 하여 주변 장치로의 데이터의 기록 처리가 종료하면, 즉 제 9 도의 단계 S12의 처리가 종료하면 단계 S14, S15로 차례로 진행하고 주변 장치로 출력되어 있던 칩 선택 신호 어드레스(SAddress)의 출력이 정지된다.

그래서 단계 S1로 되돌아가 다시 단계 S1로부터의 처리를 되풀이한다.

이상과 같이 해서 게이트웨이(4)에서는 타이밍이 제어와 버스 폭 및 어드레스 형식의 제어와 병렬로 하여(동시에) 행해지므로 장치 전체의 처리 속도를 향상시킬 수가 있는 외에 하드웨어에 대한 의존성이 적은 응용 프로그램을 제공할 수가 있게 된다.

발명의 효과

본 발명의 버스 제어 장치에 의하면 기억수단에 기억되어 있는 주변 장치의 동작 타이밍에 의거해서 중앙 연산 처리 장치와 주변 장치와의 사이의 액세스 타이밍을 제어한다. 따라서 중앙 연산 처리 장치와 주변 장치와의 사이의 액세스 타이밍을 조정하는 예를 들면 대기 컨트롤러 등의 하드웨어를 주변 장치마다 예를 들면 컨트롤러 등의 하드웨어를 주변 장치마다 설치할 필요가 없어지므로 장치 전체를 대형화 고가화 하는 일없이 중앙 연산 처리 장치와 주변 장치와의 사이의 액세스 타이밍을 제어할 수가 있다.

본 발명의 버스 제어 장치에 의하면 기억 수단에 기억된 주변 장치의 동작 타이밍이 고쳐 쓸 수가 있으므로 예를 들면 주변 장치의 사양이 변경되거나 새로운 주변 장치가 추가되어도 장치가 실질상 사용이 불가능하게 되는 것이 방치된다.

본 발명의 버스 제어 장치에 의하면 기억 수단에는 주변 장치의 최소한 유지시간 또는 복원 시간이 기억되어져 있으므로 유지 시간 또는 복원 시간에 의거한 타이밍의 제어를 할 수가 있다.

본 발명의 버스 제어 장치에 의하면 데이터 래치 수단에 의해 중앙 연산 처리 장치 또는 주변 장치가 출력하는 데이터를 래치하고 어드레스 래치 수단에 의해 중앙 연산 처리 장치가 주변 장치에 출력하는 어드레스 래치한다. 그래서 기억 수단에 기억되어 있는 주변 장치의 버스 폭에 의거해서 데이터 래치 수단으로 래치된 데이터의 팩/언팩을 제어함과 함께 기억 수단에 기억되어 있는 주변 장치의 어드레스 형식에 의거해서 어드레스 래치 수단으로 래치된 어드레스를 제어한다. 따라서 데이터의 팩/언팩 및 어드레스 형식을 고려하지 않고 응용 프로그램을 설계할 수가 있게 되어 범용성이 높은 응용 프로그램을 제공할 수가 있게 된다.

본 발명의 버스 제어 장치에 의하면 기억 수단에 기억되어 있는 주변 장치의 버스 폭과 어드레스 형식이 고쳐 쓸 수가 있으므로 예를 들면 주변 장치의 사양이 변경되거나 새로운 주변 장치가 추가되어도 장치가 실질상 사용이 불가능해지는 일이 방지된다.

(57) 청구의 범위

청구항 1.

메인 버스 및 서브 버스의 동작을 제어하는 버스 제어 장치로서, 상기 메인 버스는 중앙 처리 장치(CPU)에 결합되어 있고 상기 서브 버스는 적어도 하나의 주변 장치에 결합되어 있는, 상기 버스 제어 장치에 있어서,

상기 중앙 처리 장치, 상기 적어도 하나의 주변 장치, 상기 메인 버스, 및 상기 서브 버스의 사용을 필요로 하는 과정들의 실행을 제어하는 수단; 및

상기 적어도 하나의 주변 장치로부터 상기 서브 버스를 통해 상기 버스 제어 장치로 전송된 판독 데이터를 기억하도록 CPU 판독 동작 동안 동작 가능하며, 상기 메인 버스를 통해 상기 버스 제어 장치에 의해 수신되어 상기 서브 버스를 통해 상기 적어도 하나의 주변 장치로 전송된 기록 데이터를 기억하도록 CPU 기록 동작 동안 동작 가능한 데이터 버퍼로서, 상기 CPU 판독 동작은 상기 적어도 하나의 주변 장치로부터 몇몇 데이터가 판독될 때마다 상기 CPU에 의해 개시되며, 상기 CPU 기록 동작은 상기 적어도 하나의 주변 장치에 몇몇 데이터가 기록될 때마다 상기 CPU에 의해 개시되는, 상기 데이터 버퍼를 포함하며;

상기 제어 수단은 하나 또는 그 이상의 중앙 처리 장치 제어 신호들에 응답해서 하나 또는 그 이상의 서브 버스 제어 신호들을 발생하며,

상기 제어 수단은 상기 메인 버스 및 상기 서브 버스의 독립적인 동작을 실행할 수 있어서 임의의 소정 시간에 상기 메인 버스 및 상기 서브 버스는 상이한 과정들에 전용 가능하며,

상기 제어 수단은 상기 데이터 버퍼에 상기 판독 데이터가 기억되면 상기 판독 동작이 완료되었다는 표시를 상기 CUP에게 보냄으로써 상기 CPU는 상기 판독 데이터가 상기 버퍼에 기억된 후의 시점에서 즉시 다른 처리에 전용가능하며, 상기 제어 수단은 상기 데이터 버퍼에 상기 기록 데이터가 기억되면 상기 기록 동작이 완료되었다는 표시를 상기 CPU에게 보냄으로써 상기 CPU는 상기 기록 데이터가 상기 버퍼에 기억된 후의 시점에서 즉시 다른 처리에 전용 가능한, 버스 제어 장치.

청구항 2.

제1항에 있어서,

상기 적어도 하나의 주변 장치의 하나 또는 그 이상의 동작 특성들을 기억하는 수단; 및

상기 하나 또는 그 이상의 주변 장치가 대체되거나 변경될 때, 또는 부가의 주변 장치가 상기 서브 버스에 결합될 때, 상기 기억 수단에 기억된 상기 동작 특성들을 편집하는 수단을 더 포함하는 버스 제어 장치.

청구항 3.

제1항에 있어서,

상기 적어도 하나의 주변 장치의 하나 또는 그 이상의 동작 특성들을 기억하는 수단을 더 포함하며,

상기 동작 특성들은 유지 시간(hold time)을 포함하는, 버스 제어 장치.

청구항 4.

제1항에 있어서,

상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭보다 더 큰, 버스 제어 장치.

청구항 5.

제1항에 있어서,

상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭의 정수 배와 동일한, 버스 제어 장치.

청구항 6.

중앙 처리 장치와 복수의 상이한 주변 장치들이 접속되는 복수의 버스 라인들을 상호 접속하는 버스 제어 장치에 있어서,

적어도 중앙 처리 장치가 접속되는 메인 버스;

상기 복수의 상이한 주변 장치들이 접속되는 서브 버스;

상기 복수의 상이한 주변 장치들의 버스 폭과 어드레스 형식 정보를 기억하는 기억 수단으로서, 상기 어드레스 형식 정보는 각각의 주변 장치가 고정 어드레스를 활용하는지 가변 어드레스를 활용하는지를 표시하는, 상기 기억 수단; 및

상기 기억 수단에 기억된 상기 각각의 주변 장치의 각각의 버스 폭과 각각의 어드레스 형식에 따라 상기 중앙 처리 장치에 의해 상기 각각의 주변 장치로부터 데이터를 판독하는 것과 상기 각각의 주변 장치에 데이터를 기록하는 것을 제어하는 단일의 제어기를 포함하는 버스 제어 장치.

청구항 7.

제6항에 있어서,

상기 기억 수단에 기억된 상기 복수의 주변 장치들의 상기 버스 폭 및 상기 어드레스 형식 정보는 상기 복수의 주변 장치들의 동작 명세(operating specifications)가 변경되거나 새로운 주변 장치가 부가될 때 재 기록되며, 이에 의해 상기 중앙 처리 장치의 여기 애플리케이션 소프트웨어(exciting application software)는 임의의 변형 없이 동작 가능한, 버스 제어 장치.

청구항 8.

제7항에 있어서,

상기 기억 수단은 상기 복수의 주변 장치들의 적어도 유지 시간 또는 복원 시간을 기억하도록 동작가능하며, 상기 단일 제어기는 상기 기억 수단에 기억된 상기 각각의 주변 장치의 각각의 유지 시간 또는 복원 시간에 따라 상기 각각의 주변 장치로부터 데이터를 판독하는 것을 제어하도록 동작가능하고 상기 각각의 주변 장치에 데이터를 기록하는 것을 제어하도록 동작 가능한, 버스 제어 장치.

청구항 9.

제8항에 있어서,

상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭보다 더 큰, 버스 제어 장치.

청구항 10.

제9항에 있어서,

상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭의 정수 배와 동일한, 버스 제어 장치.

청구항 11.

제1항에 있어서,

상기 적어도 하나의 주변 장치의 하나 또는 그 이상의 동작 특성들을 기억하는 수단을 더 포함하며,

상기 동작 특성들은 복원 시간(recovery time)을 포함하는, 버스 제어 장치.

청구항 12.

메인 버스 및 서브 버스의 동작을 제어하는 버스 제어 방법으로서, 상기 메인 버스는 중앙 처리 장치(CPU)에 결합되어 있고 상기 서브 버스는 적어도 하나의 주변 장치에 결합되어 있는, 상기 버스 제어 방법에 있어서,

하나 또는 그 이상의 중앙 처리 장치 제어 신호들에 응답해서 하나 또는 그이상의 서브 버스 제어 신호들을 발생함으로써 상기 중앙 처리 장치, 상기 적어도 하나의 주변 장치, 상기 메인 버스 및 상기 서브 버스의 사용을 필요로 하는 과정들의 실행을 제어하는 단계를 포함하며,

CPU 판독 동작이 실행될 때, 상기 적어도 하나의 주변 장치로부터 상기 서브 버스를 통해 버스 제어 장치로 전송되는 판독 데이터를 중간 매체 데이터 버퍼에 기억하고, CPU 기록 동작이 실행될 때, 상기 메인 버스를 통해 상기 버스 제어 장치에 의해 수신되어 상기 서브 버스를 통해 상기 적어도 하나의 주변 장치로 전송되는 기록 데이터를 상기 중간 매체 데이터 버퍼에 기억하며, 상기 CPU 판독 동작은 상기 적어도 하나의 주변 장치로부터 몇몇 데이터가 판독될 때마다 상기 CPU에의해 개시되며, 상기 CPU 기록 동작은 상기 적어도 하나의 주변 장치에 몇몇 데이터가 기록될 때마다 상기 CPU에 의해 개시되며,

상기 중간 매체 데이터 버퍼에 상기 판독 데이터를 기억하면 상기 판독 동작이 완료되었다는 표시를 상기 CPU에 보냄으로써 상기 CPU는 상기 판독 데이터가 상기 버퍼에 기억된 후의 시점에서 즉시 다른 처리에 전용가능하며, 상기 중간 매체 데이터 버퍼에 상기 기록 데이터를 기억하면 상기 기록 동작이 완료되었다는 표시를 상기 CPU에 보냄으로써 상기 CPU는 상기 기록 데이터가 상기 버퍼에 기억된 후의 시점에서 즉시 다른 처리에 전용 가능한, 버스 제어 방법.

청구항 13.

제12항에 있어서,

상기 적어도 하나의 주변 장치의 하나 또는 그 이상의 동작 특성들을 기억하는 단계; 및

상기 하나 또는 그 이상의 주변 장치가 대체되거나 변경될 때, 또는 부가의 주변 장치가 상기 서브 버스에 결합될 때, 상기 기억된 동작 특성들을 편집하는 단계를 더 포함하는 버스 제어 방법.

청구항 14.

제12항에 있어서,

상기 적어도 하나의 주변 장치의 하나 또는 그 이상의 동작 특성들을 기억하는 단계를 더 포함하며,

상기 동작 특성들은 유지 시간을 포함하는, 버스 제어 방법.

청구항 15.

제12항에 있어서,

상기 적어도 하나의 주변 장치의 하나 또는 그 이상의 동작 특성들을 기억하는 단계를 더 포함하며,

상기 동작 특성들은 복원 시간을 포함하는, 버스 제어 방법.

청구항 16.

제12항에 있어서,

상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭보다 더 큰, 버스 제어 방법.

청구항 17.

제12항에 있어서,

상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭의 정수 배와 동일한, 버스 제어 방법.

청구항 18.

중앙 처리 장치와 복수의 상이한 주변 장치들이 접속되는 복수의 버스 라인들을 상호 접속시키는 방법에 있어서,

적어도 상기 중앙 처리 장치를 메인 버스에 접속시키는 단계;

상기 복수의 상이한 주변 장치들을 서브 버스에 접속시키는 단계;

상기 복수의 상이한 주변 장치들의 버스 폭과 어드레스 형식 정보를 기억하는 단계로서, 상기 어드레스 형식 정보는 각각의 주변 장치가 고정 어드레스를 활용하는지 가변 어드레스를 활용하는지를 표시하는, 상기 기억 단계; 및

상기 기억 수단에 기억된 상기 각각의 주변 장치의 상기 각각의 버스 폭 및 상기 각각의 어드레스 형식에 따라 상기 중앙 처리 장치에 의해 상기 각각의 주변 장치로부터 데이터를 판독하는 것을 제어하고 상기 각각의 주변 장치에 데이터를 기록하는 것을 제어하는 단계를 포함하는 상호접속 방법.

청구항 19.

제18항에 있어서,

상기 복수의 주변 장치들의 동작 명세가 변경될 때 또는 새로운 주변 장치가 부가될 때, 상기 복수의 상이한 주변 장치들의 상기 버스 폭 및 상기 어드레스 형식 정보는 재 기록되며, 이에 의해 상기 중앙 처리 장치의 여기 애플리케이션 소프트웨어는 임의의 변형 없이 동작 가능한, 상호접속 방법.

청구항 20.

제19항에 있어서,

상기 복수의 상이한 주변 장치들의 적어도 유지 시간 또는 복원 시간을 기억하는 단계와, 상기 각각의 주변 장치의 상기 각각의 유지 시간 또는 복원 시간에 따라 상기 각각의 주변 장치로부터 데이터를 판독하는 것을 제어하고 상기 각각의 주변 장치에 데이터를 기록하는 것을 제어하는 단계를 더 포함하는 상호접속 방법.

청구항 21.

제20항에 있어서,

상기 메인 버스는 상기 서브 버스의 버스 폭보다 더 큰 버스 폭을 갖는, 상호접속 방법.

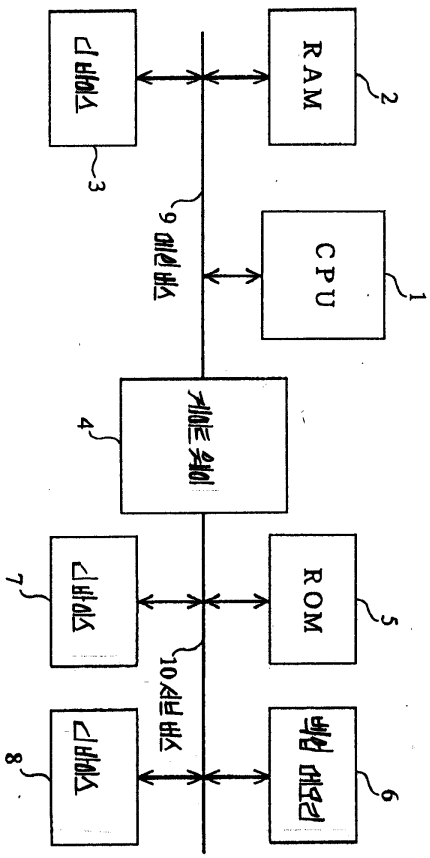
청구항 22.

제21항에 있어서,

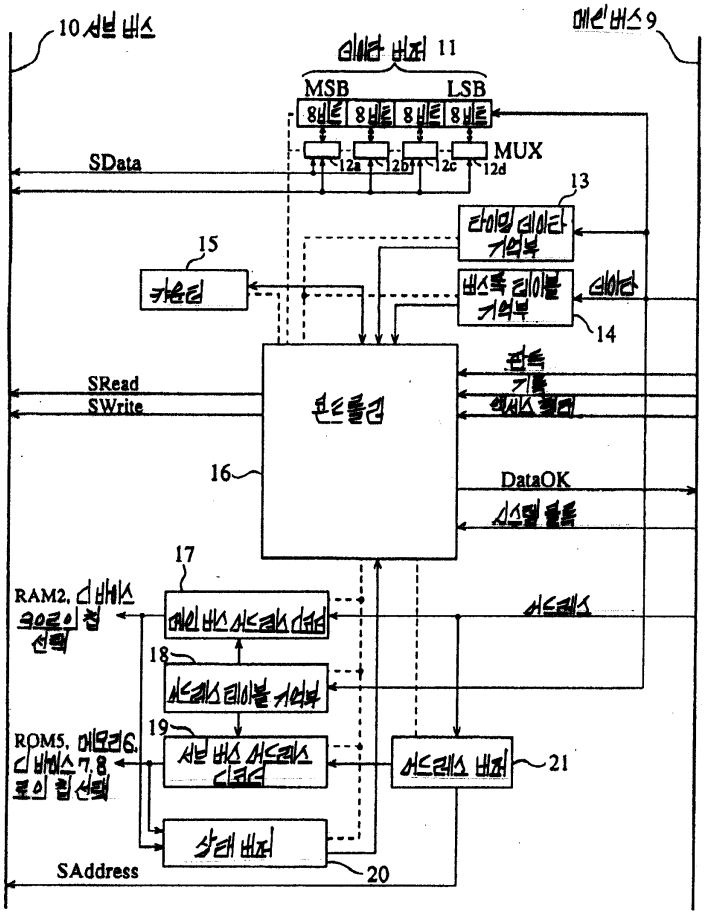
상기 메인 버스의 버스 폭은 상기 서브 버스의 버스 폭의 정수 배와 동일한, 상호접속 방법.

도면

도면 1



도면 2



게이트웨이 4

도면 3

(단위 : 클록)

	D 버퍼	D 버퍼 8	ROM5	메모리 6
컨트 지연 시간	2	3	2	4
데이 지연 시간	1	3	—	4
유지 시간	1	1	0	0
플로팅 시간	2	0	0	0
복원 시간	0	0	0	10

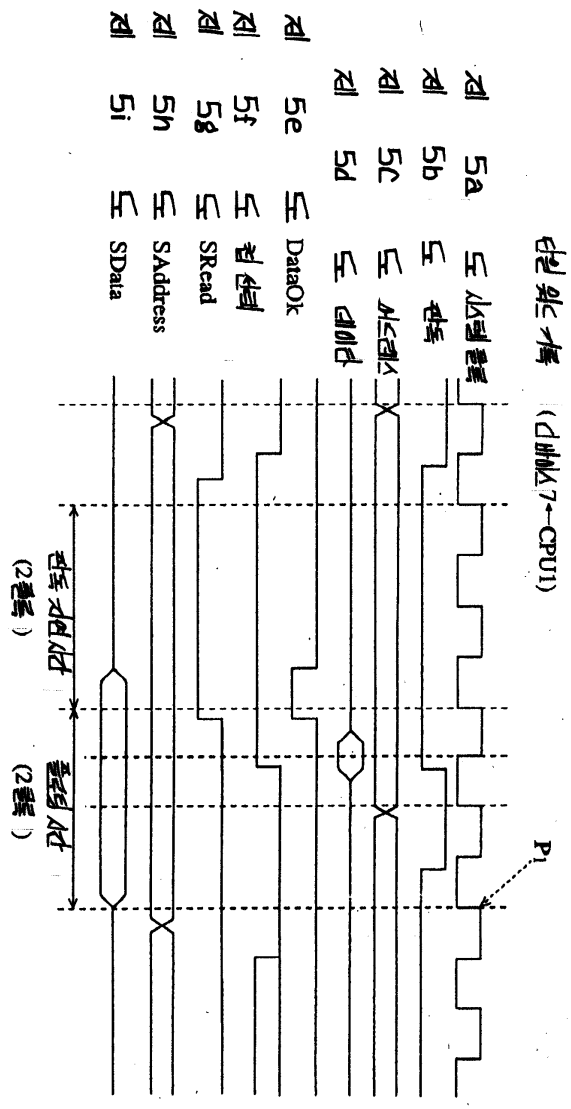
타이밍 테이블

도면 4

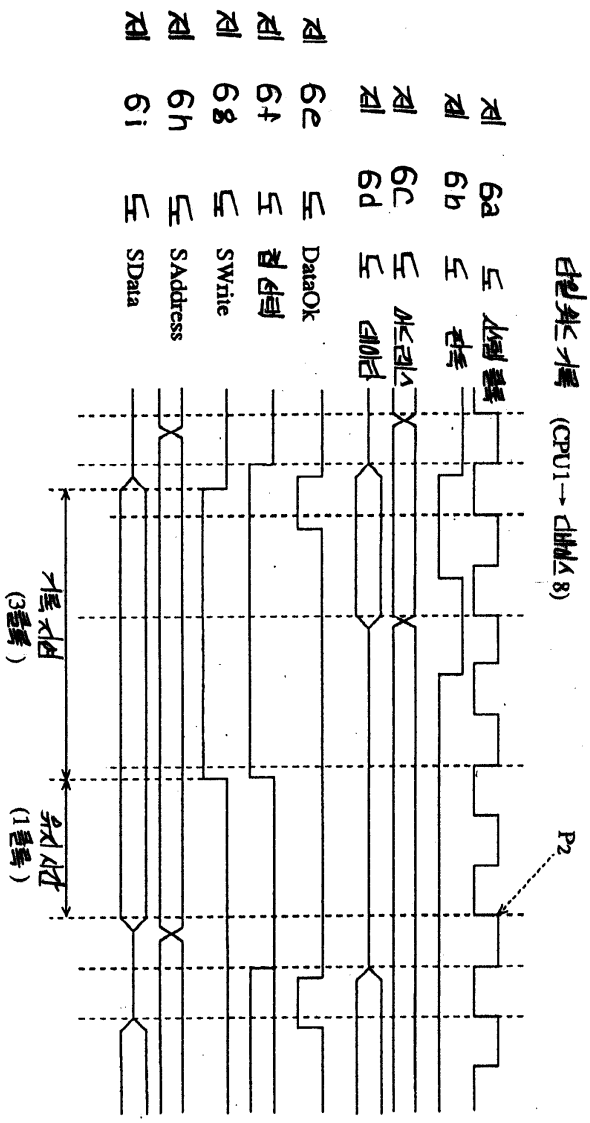
	디바이스 7	디바이스 8	ROM5	메모리6
버스폭	8 비트	16 비트	8 비트	16비트
어드레스 형식	고정	고정	가변	가변

룩업테이블유닛

도면 5

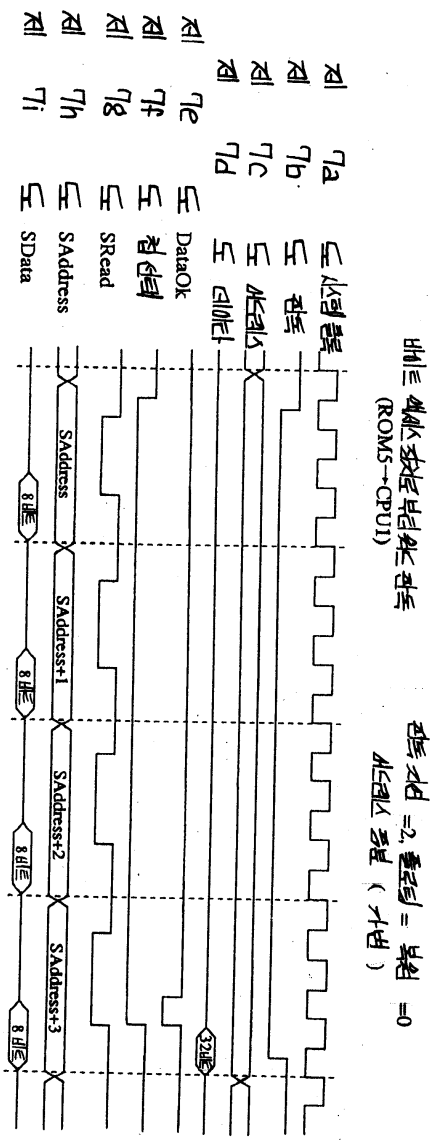


도면 6



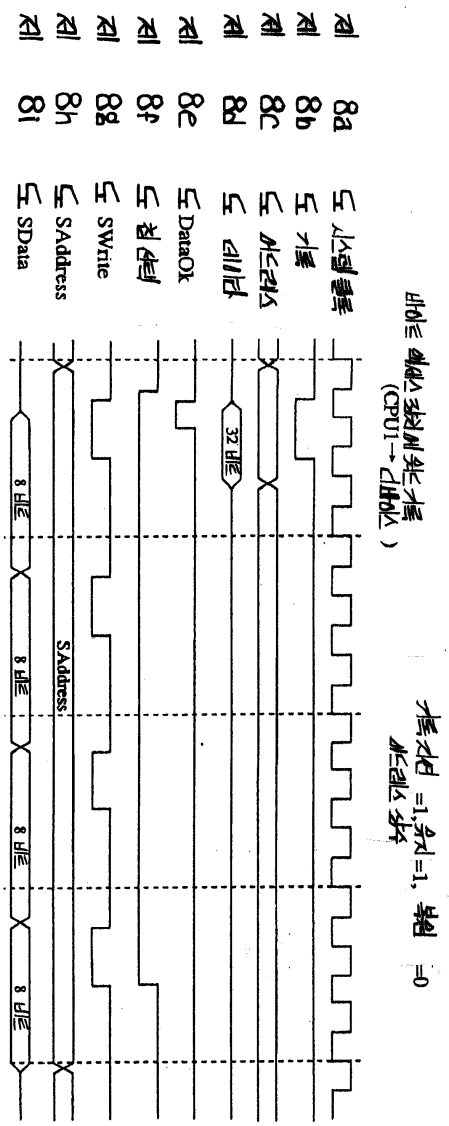
단일 웨스 기록 (CPU1→ 대비스 8)
제 6a 도 선택 클록
제 6b 도 전속
제 6c 도 어드레스
제 6d 도 데이터
제 6e 도 DataOk
제 6f 도 칩 선택
제 6g 도 SWrite
제 6h 도 SAddress
제 6i 도 SData
기록 지연 (3클록)
유지 시간 (1클록)
P2

도면 7

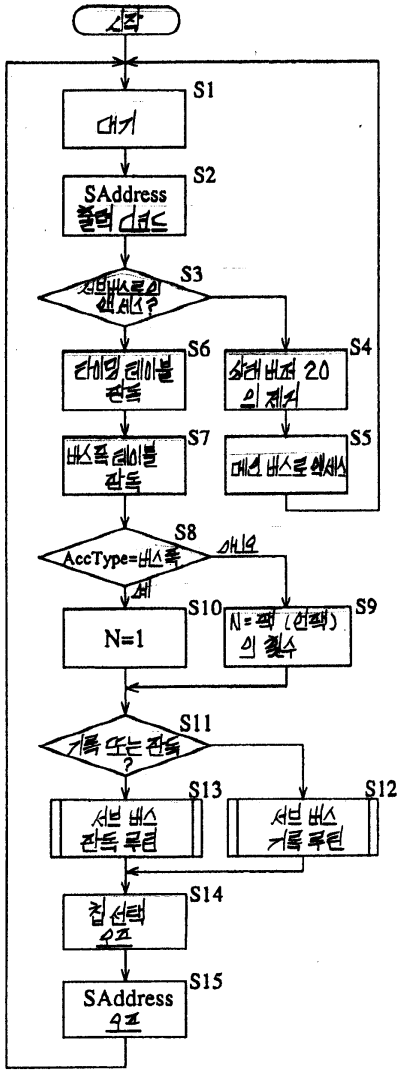


비이트 액세스 장치로 부터 왓드 전독
(ROM5→CPU1)
전독 지연 =2, 홀로딩 = 부원 =0
사드레스 증분 (가변)
제 7a 도 사용 클록
제 7b 도 전독
제 7c 도 사드레스
제 7d 도 데이타
제 7e 도 DataOk
제 7f 도 첩선택
제 7g 도 SRead
제 7h 도 SAddress
제 7i 도 SData
SAddress
SAddress+1
SAddress+2
SAddress+3
8비트
32비트

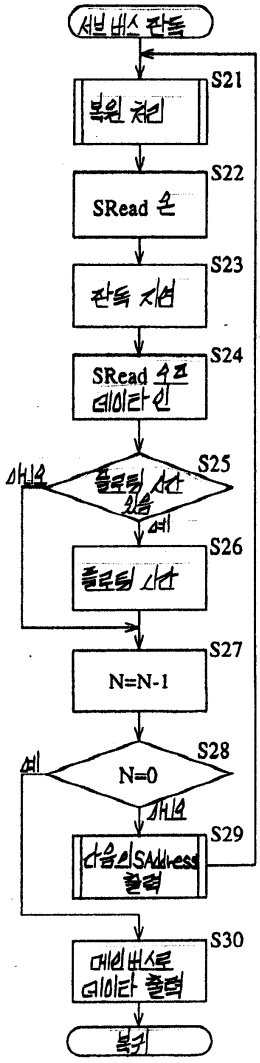
도면 8


도 시스템 클록
도 기록
도 어드레스
도 데이터
도 DataOk
도 쓰기 신호
도 SWrite
도 SAddress
도 SData
8a
8b
8c
8d
8e
8f
8g
8h
8i
32 비트
8 비트
8 비트
8 비트
8 비트
SAddress

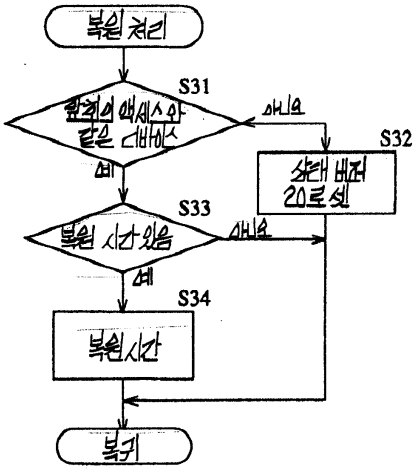
도면 9


시작
S1
대기
S2
SAddress
S3
S6
S4
S7
S5
S8
아니오
예
AccType=버스록
S10
N=1
S9
S11
S13
S12
서브 버스
서브 버스
S14
S15
SAddress

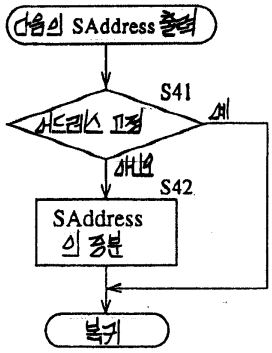
도면 10


서브 버스 관리
S21
복원 처리
S22
SRead 온
S23
관리 지연
S24
SRead 수프 데이타 인
S25
플로팅 서브 있음
아니요
예
S26
플로팅 서브
S27
N=N-1
S28
N=0
예
아니요
S29
다음의 SAddress 출력
S30
메인 버스로 데이타 출력
복귀

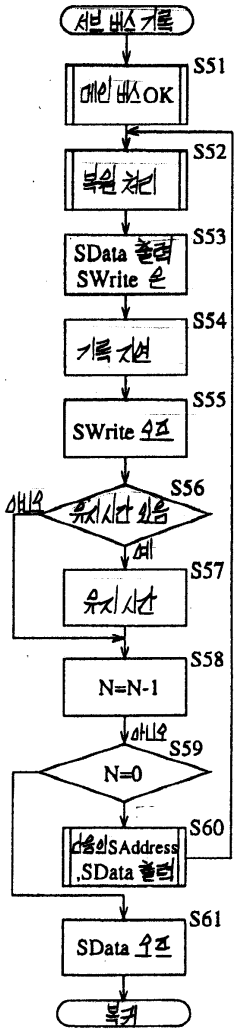
도면 11


복원 처리
S31
현재의 액세스와 같은 디바이스
아니요
예
S32
상태 버퍼 20로 set
S33
복원 시간 있음
아니요
예
S34
복원시간
복귀

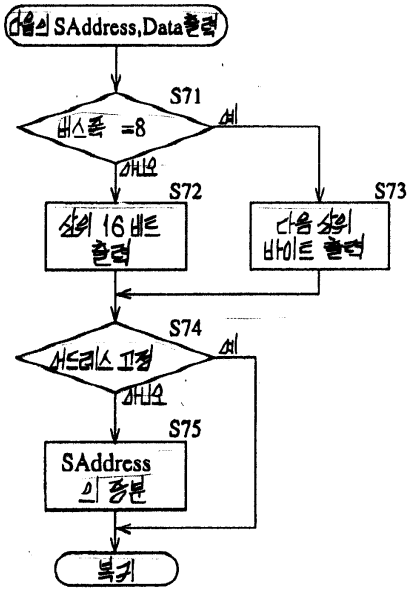
도면 12


다음의 SAddress 출력
S41
어드레스 고정
예
아니요
S42
SAddress 의 증분
복귀

도면 13


서브 버스 기록
S51
메인 버스OK
S52
복원 처리
S53
SData 출력
SWrite 온
S54
기록 지연
S55
SWrite 오프
S56
유지시간 있음
아니오
예
S57
유지 시간
S58
N=N-1
아니오
S59
N=0
S60
SAddress
,SData 출력
S61
SData 오프
복귀

도면 14


다음의 SAddress,Data출력
S71
버스폭 =8
예
아니오
S72
상위 16 비트 출력
S73
다음 상위 바이트 출력
S74
어드레스 고정
예
아니오
S75
SAddress 의 증분
복귀