[JO2122] 컴퓨터구조

2022년 1학기

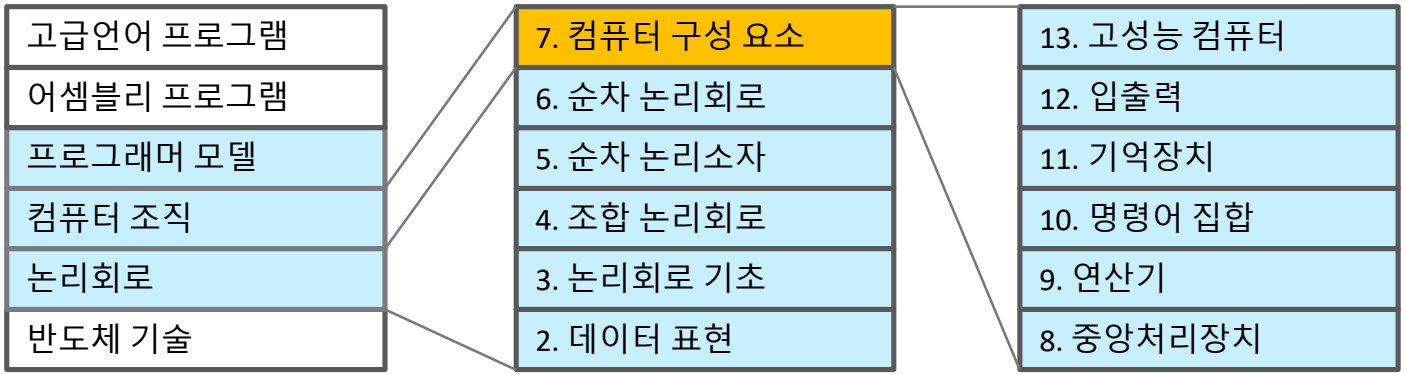
상명대학교 소프트웨어학과 박희민

7.1 프로그램 실행
7.2 컴퓨터 구성 요소
7.3 시스템 버스
7.4 명령어
7.5 요약

2022-04-27

CHAP07 컴퓨터구성요소

제7장 컴퓨터 구성 요소



- 학습 목표
  - 프로그램 내장형 컴퓨터가 프로그램을 실행하는 방법을 이해한다.
  - 중앙처리장치, 기억장치, 입출력장치의 기능을 설명할 수 있다.
- 내용

7.1 프로그램 실행
7.2 컴퓨터 구성 요소
7.3 시스템 버스
7.4 명령어
7.5 요약

7.1 프로그램의 실행

- 프로그램의 실행
  - 프로그램은 기억장치에 차례대로 적재되어 있는 기계어 명령어
  - 명령어 사이클 {인출단계, 실행단계} 반복
- 학습 목표
  - 프로그램 내장형 (stored program) 컴퓨터가 프로그램을 실행하는 방법을 설명할 수 있다.

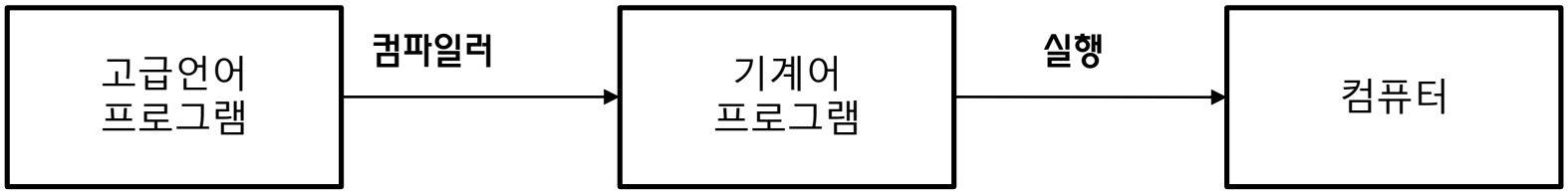
- 내용

7.1.1 기계어 프로그램

7.1.2 프로그램 내장형 컴퓨터

7.1.1 기계어 프로그램

- 컴퓨터
  - 프로그램을 실행하는 기계
  - 프로그램은 기계어(machine instructions)의 모임
  - 기계어 명령어를 실행하는 기계
- 기계어 (machine language)
  - 2진수 코드로 표현한 명령어.
  - 명령어 종류별로 길이가 다를 수 있다.
- 어셈블리 언어(assembly language)
  - 의미를 쉽게 알 수 있도록 기호(니모닉 코드)로 표현한 명령어
  - 어셈블러는 어셈블리 언어 명령어를 기계어로 번역한다.



※ sequence of instructions

Pentium 프로세서 기계어 프로그램

```
1.  int n, sum;
2.  n = 1;
3.  sum = 0;
4.  while (n <= 100) {
5.      sum = sum + n;
6.      n = n + 1;
7.  }
```

번호	주소	기계어 코드	어셈블리 언어	
1.	0041138E	c745f800000001	mov	dword ptr [n],1
2.	00411395	c745ec00000000	mov	dword ptr [sum],0
3.	0041139C	837df864	cmp	dword ptr [n],64h
4.	004113A0	7f14	jg	wmain+46h (4113B6h)
5.	004113A2	8b45ec	mov	eax, dword ptr [sum]
6.	004113A5	0345f8	add	eax, dword ptr [n]
7.	004113A8	8945ec	mov	dword ptr [sum], eax
8.	004113AB	8b45f8	mov	eax, dword ptr [n]
9.	004113AE	83c001	add	eax, 1
10.	004113B1	8945f8	mov	dword ptr [n],eax
11.	004113B4	ebe6	jmp	wmain+2Ch (41139Ch)

(b) Pentium 프로세서의 기계어 프로그램

- ※ Pentium 기계어 코드는 명령어 종류마다 길이가 다르다.
- ※ 명령어는 기억장치에 차례대로 적재되어 있다.

AVR 기계어 프로그램

```
1.  int n = 1;
2.  int sum = 0;

3.  while (n <= 100) {
4.      sum = sum + n;
5.      n = n + 1;
6.  }
```

번호	주소	기계어 코드		어셈블리 언어	
1.	000067	e001		LDI	R16, LOW(1)
2.	000068	e010		LDI	R17, HIGH(1)
3.	000069	e020		LDI	R18, LOW(0)
4.	00006a	e030		LDI	R19, HIGH(0)
5.	00006b	3605	_0x3:	CPI	R16, LOW(101)
6.	00006c	e0e0		LDI	R30, HIGH(101)
7.	00006d	071e		CPC	R17, R30
8.	00006e	f42c		BRGE	_0x5
9.	00006f	0f20		ADD	R18, R16
10.	000070	1f31		ADC	R19, R17
11.	000071	5f0f		SUBI	R16, LOW(-1)
12.	000072	4f1f		SBCI	R17, HIGH(-1)
13.	000073	cff7		RJMP	_0x3
14.	000074		_0x5:		

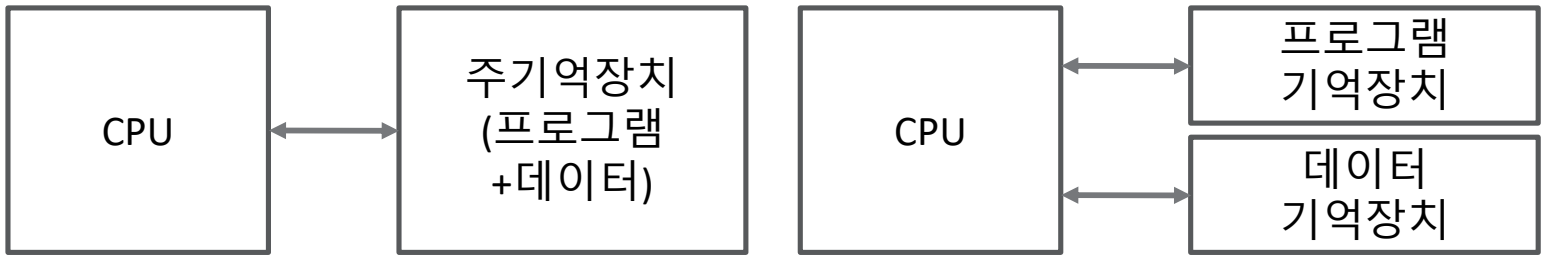
(c) AVR 프로세서의 기계어 프로그램

- ※ AVR 기계어 코드는 길이가 같다.
- ※ 기계어 프로그램은 프로세서 종류마다 다르다.

7.1.2 프로그램 내장형 컴퓨터

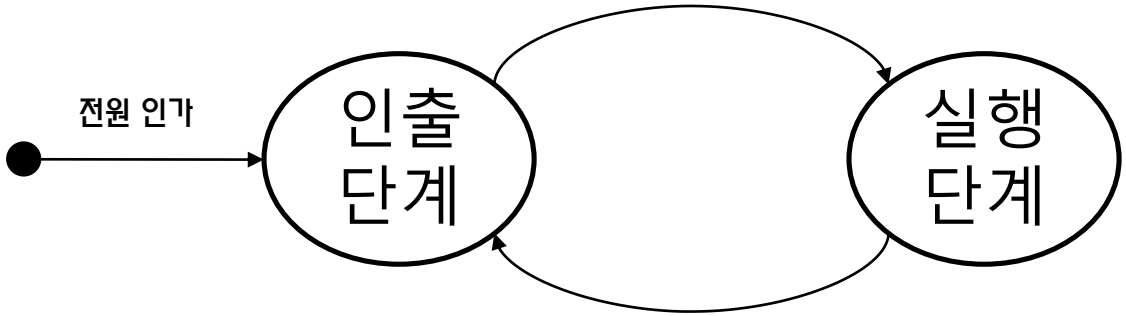
- 프로그램 내장형(stored program) 컴퓨터
  - Von Neumann 형 컴퓨터
  - 프로그램과 데이터를 주기억장치에 저장한다.
  - 명령어를 한 개씩 중앙처리장치로 가져와 실행한다.

- 기억장치 구조
  - 일반 구조: 한 개의 기억장치에 프로그램과 데이터를 함께 저장한다.
  - 하버드 구조: 프로그램과 데이터를 별도의 기억장치에 저장한다.



(a) **프로그램 내장형 컴퓨터** (b) **하버드 구조**

명령어 사이클



- 명령어 사이클 = 머신 사이클 = {인출 단계, 실행 단계}
- 인출 단계(fetch cycle)
  - 주기억장치에서 명령어를 중앙처리장치로 가져온다.
- 실행 단계(execution cycle)
  - 중앙처리장치에서 명령어를 해독하고 실행한다.

7.2 컴퓨터 구성 요소

- 학습 목표
  - 컴퓨터의 세 가지 구성 요소인 중앙처리장치, 주기억장치, 입출력 장치의 기능을 이해한다.

- 내용

7.2.1 중앙처리장치

7.2.2 주기억장치

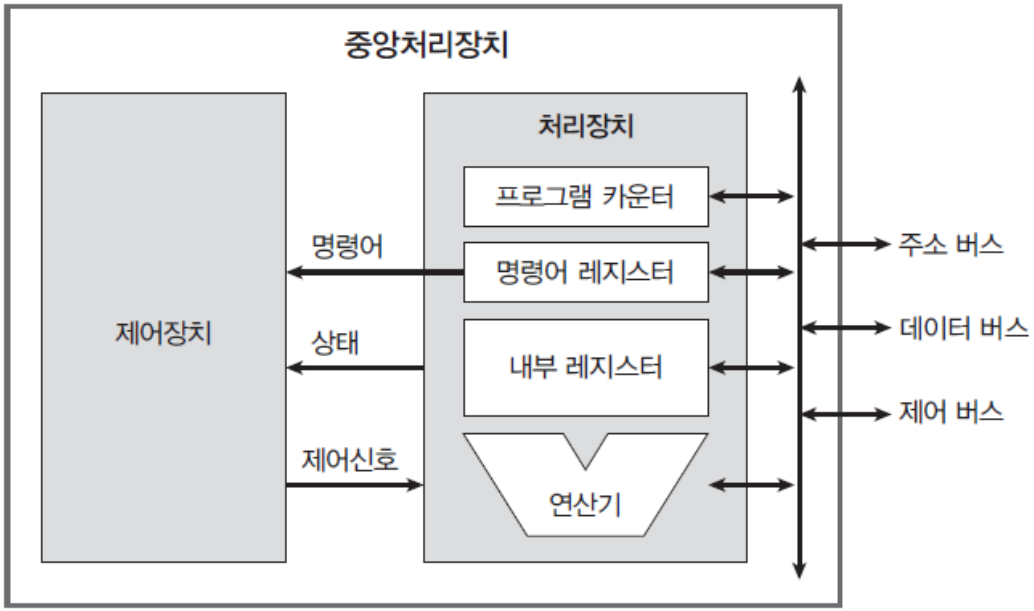
7.2.3 입출력 장치

7.2.4 AVR 마이크로제어기

7.2.1 중앙처리장치

- 중앙처리장치 기능
  - 명령어 사이클을 반복 실행하여 프로그램을 실행한다.
- 인출 단계
  - 필요한 구성 요소
    - PC (Program Counter): 다음에 실행할 명령어가 들어있는 기억장소 주소를 저장한다.
    - IR (Instruction Register): 현재 실행하고 있는 명령어를 저장한다.
  - 동작: 기억장치에서 명령어를 하나씩 가져온다.
    - IR ← Mem[PC], PC ← PC + [명령어의 길이]
- 실행 단계
  - 필요한 구성 요소
    - 내부 레지스터 (internal registers): 처리할 데이터를 임시 저장한다.
    - 연산기(ALU, Arithmetic and Logic Unit): 데이터 연산을 담당한다.
  - 동작
    - IR의 명령어를 해석하여 실행한다.
    - 명령어마다 수행하는 일이 다르다.

중앙처리장치 내부 구조



[그림 7-4] 중앙처리장치 내부 구조

- 제어장치
  - IR의 명령어 코드 해석
  - 중앙처리장치 내부와 외부 제어신호 생성
- 처리장치
  - PC: 다음에 실행할 명령어의 주소 저장
  - IR: 현재 실행 중인 명령어 저장
  - 내부 레지스터: 데이터 임시 저장
  - 연산기: 데이터 조작(산술연산, 논리연산)
- 버스
  - 내부 버스
  - 외부 버스: 주소 버스, 데이터 버스, 제어 버스

7.2.1 중앙처리장치

중앙처리장치 기능

- 명령어를 실행한다.
- 구성 요소를 제어하여 명령어 사이클 반복 실행

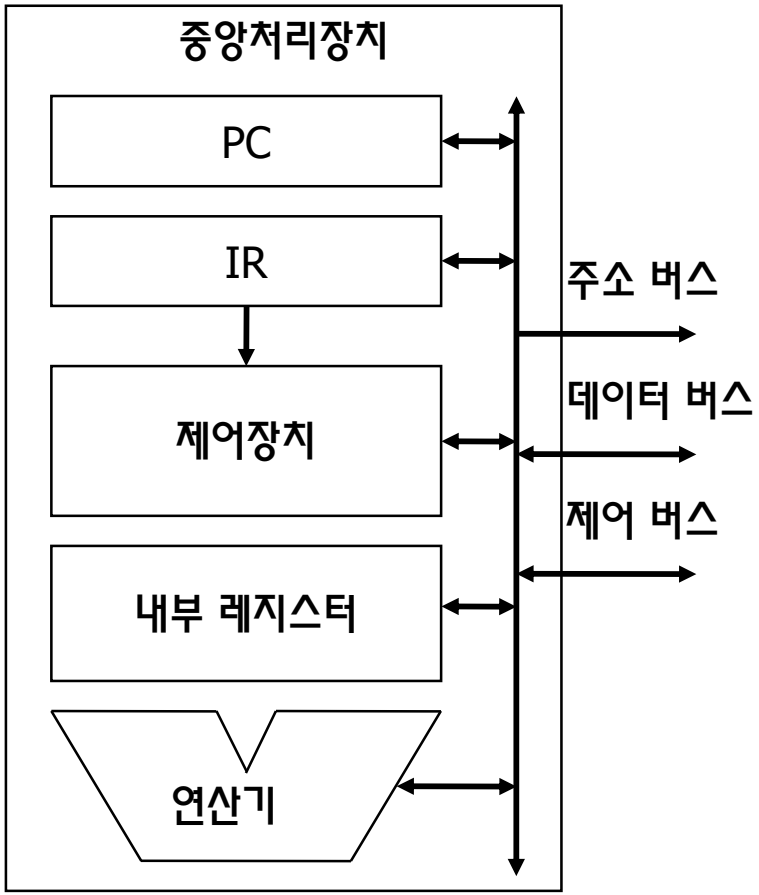
인출 단계

- 기억장치에서 명령어를 하나씩 가져온다.
- 필요한 구성 요소
  - PC (Program Counter)
  - IR (Instruction Register)
- 동작
  - IR ← Mem[PC],
    PC ← PC + [명령어의 길이]

실행 단계

- IR의 명령어를 해석하여 실행한다.
- 필요한 구성 요소
  - 내부 레지스터 (internal registers)
  - 연산기(ALU, Arithmetic and Logic Unit)
- 동작
  - 명령어마다 수행하는 일이 다르다.

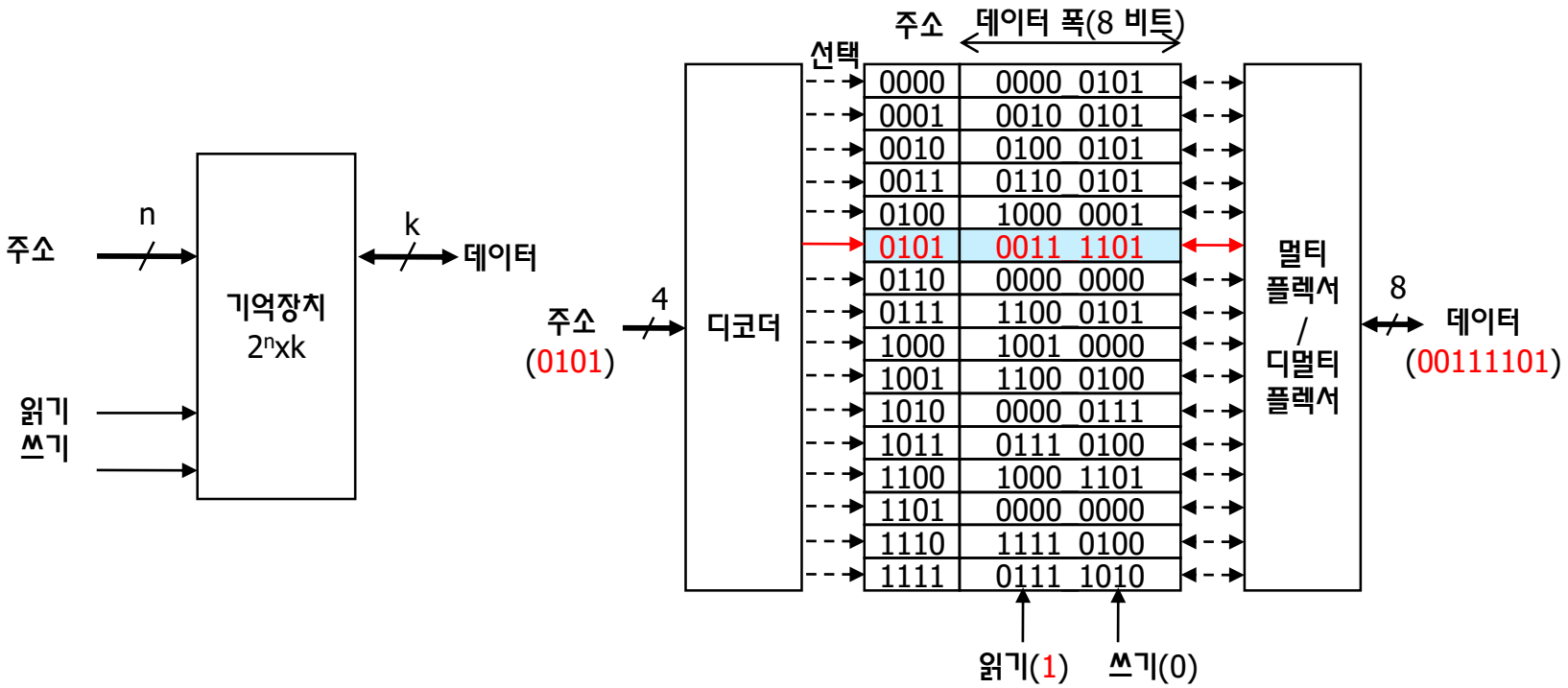
중앙처리장치 구조



- 제어장치
  - IR의 명령어 코드 해석
  - 중앙처리장치 내부와 외부 제어신호 생성
- 처리장치
  - PC: 다음에 실행할 명령어의 주소 저장
  - IR: 현재 실행 중인 명령어 저장
  - 내부 레지스터: 데이터 임시 저장
  - 연산기: 데이터 조작(산술연산, 논리연산)
- 버스
  - 내부 버스
  - 외부 버스: 주소버스, 데이터버스, 제어버스

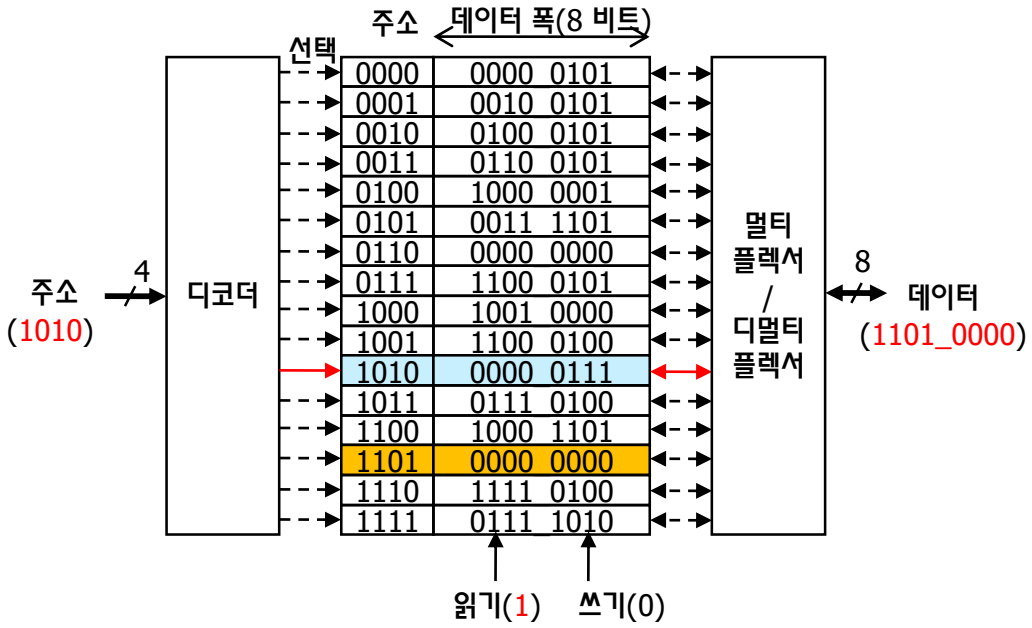
7.2.2 주기억장치

- 주기억장치: 프로그램과 데이터를 저장하는 1차원 배열
- 기억장치 용량: 2^n x k 비트 (n비트 주소 선, k비트 데이터 선)
- 적재(load): Register ← Mem(address) # 기억장치 읽기
- 저장(store): Mem(address) ← Register # 기억장치 쓰기
- 단어(word): 중앙처리장치가 한 번에 액세스하는 데이터



[예제]

- [예제 7-1] 기억장치 동작
  1) 주소선=1010, 읽기=0, 쓰기=1, 데이터선=1101_0000일 때, 기억장치 동작은?
  2) 주소선=1101, 읽기=1, 쓰기=0일 때, 데이터 선의 값은?



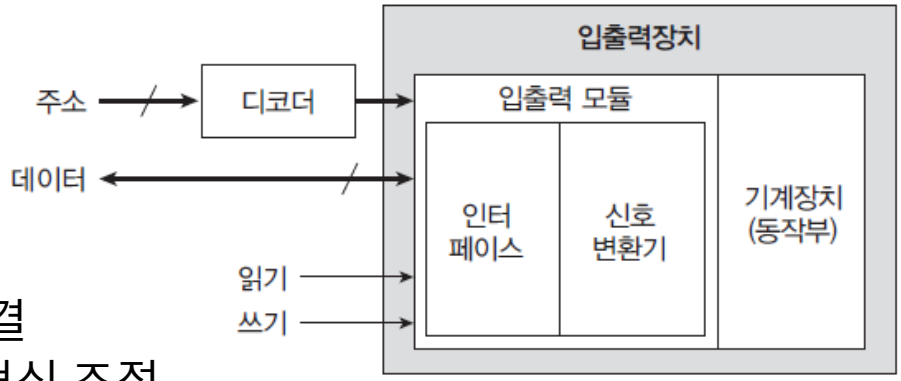
- [예제 7-2] 기억장치 주소선 수와 데이터선 수는?
  1) 16M×16비트 주소선 = _____ 비트, 데이터선 = _____비트
  2) 4G×8비트 주소선 = _____ 비트, 데이터선 = _____비트

7.2.3 입출력장치

- 기능
  - 프로그램과 데이터를 공급한다.
  - 컴퓨터가 처리한 데이터를 출력한다.
- 특징
  - 종류가 다양하고, 기계 동작을 포함한다.
  - 중앙처리장치와 주기억장치에 비하여 동작 속도가 느리다.
- 예
  - 입력장치: 키보드, 마우스, 마이크, 카메라, 터치 스크린, 스캐너
  - 출력장치: 모니터, 프린터, 플로터, 스피커
  - 입출력장치: 보조 기억장치(디스크, USB 메모리 등), 네트워크 어댑터

입출력장치 구조

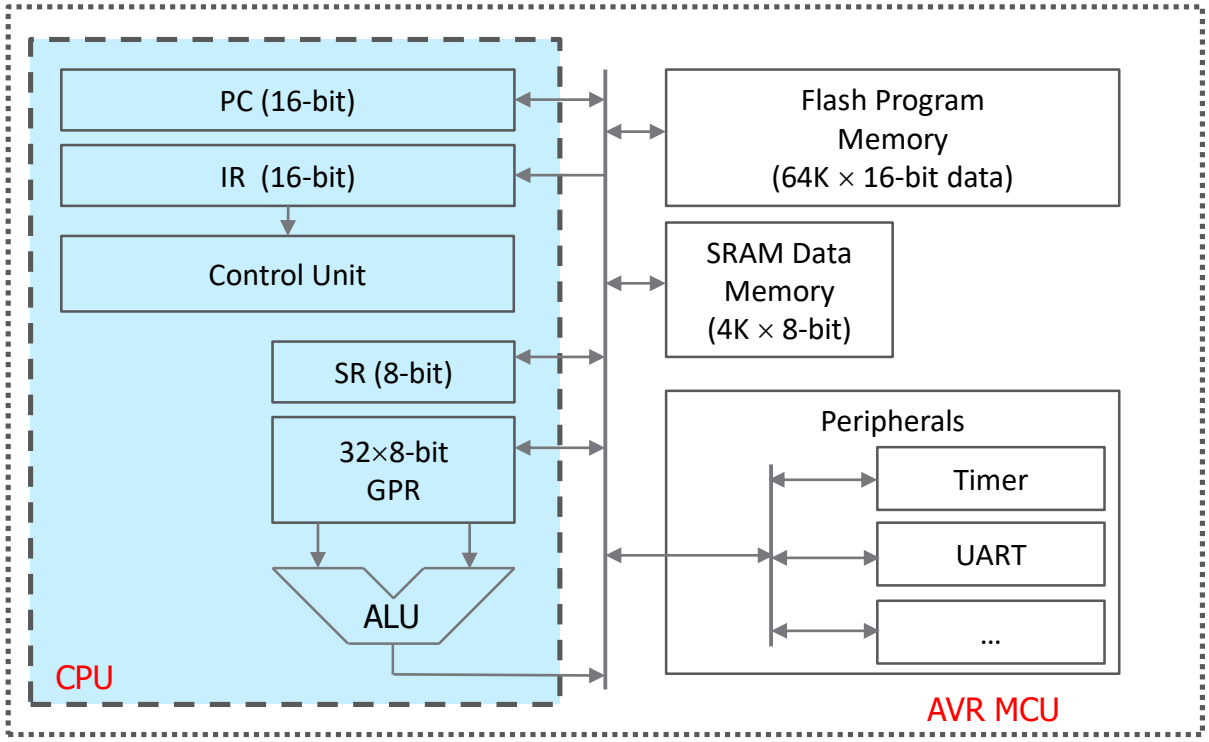
- 프로세서 연결
  - 입출력 주소(입출력 포트)
    - 입출력 장치마다 고유 주소 할당.
  - 기억장치 인터페이스와 동일
    - 주소선
    - 데이터선
    - 제어선: 읽기,쓰기
- 입출력장치 구조
  - 인터페이스(interface)
    - 프로세서와 입출력장치 연결
    - 데이터 전송 속도, 데이터 형식 조정
  - 신호변환기
    - 전기 신호 ↔ 기계 동작
  - 동작부(actuator)
    - 기계 구동



[그림 7-6] 입출력장치 구조

7.2.4 AVR 마이크로제어기

- 마이크로제어기
  - 하나의 칩에 중앙처리장치, 기억장치, 주변장치를 포함한 반도체 소자
  - 소형 임베디드 시스템의 제어장치로 쓰임.
- AVR 마이크로제어기: 아두이노 제어기



7.2 컴퓨터 구성 요소 요약

- 중앙처리장치
  - {제어장치, 처리장치(레지스터, 연산기)}
  - 프로그램 카운터(PC): 다음에 실행할 명령어가 저장된 기억장소 주소 저장
  - 명령어 레지스터(IR): 현재 실행 중인 명령어 저장
- 주기억장치
  - 기억장소를 많이 포함하고 있는 1차원 배열
  - 주소 지정에 의한 액세스
- 입출력장치
  - 입출력 포트: 입출력장치에게 할당되는 주소
  - 사용 측면에서 기억장치와 동일

7.3 시스템 버스

- 학습 목표
  - 컴퓨터의 세 가지 구성 요소를 연결하는 시스템 버스의 구성과 제어 신호의 종류를 이해한다.

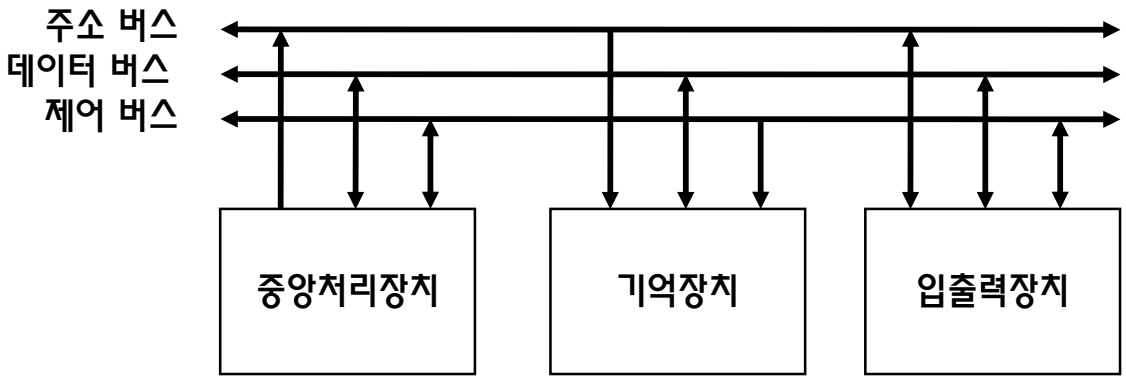
- 내용

7.3.1 시스템 버스 구성

7.3.2 제어 신호 종류

7.3.3 버스 계층

7.3.1 시스템 버스 구성



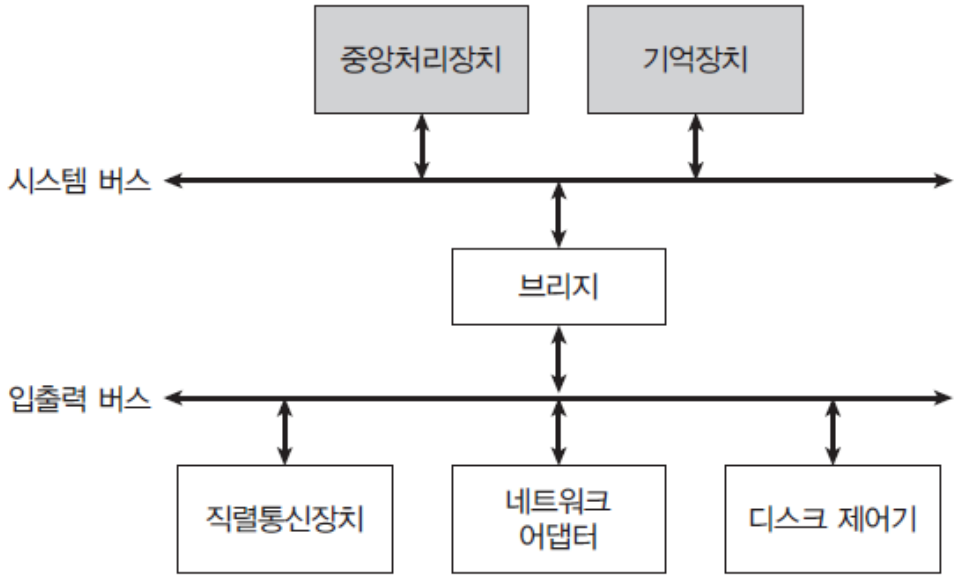
- 주소 버스
  - 기억장치 또는 입출력장치의 주소를 지정한다.
  - 기억장치 용량과 관련이 있다.
- 데이터 버스
  - 데이터를 전달하는 경로이고, 일반적으로 8의 배수이다.
  - 데이터 버스 비트 수 = 레지스터 크기 = 단어 크기(word size)
- 제어 버스
  - 주소 버스와 데이터 버스를 제외한 나머지 신호.
  - 각각 고유한 기능을 갖는다.

7.3.2 제어 신호 종류

- 기억장치 제어
  - 기억장치 읽기(memory read)
  - 기억장치 쓰기(memory write)
- 입출력장치 제어
  - 입출력 읽기(input 또는 I/O read)
  - 입출력 쓰기(output 또는 I/O write)
  - 인터럽트 요청(interrupt request)
  - 인터럽트 확인(interrupt acknowledge)
  - 버스 요청(bus request)
  - 버스 승인(bus grant)
- 기타
  - 시스템 클럭(system clock)
  - 리셋(reset)
  - 전력선(power lines)

7.3.3 버스 계층

- 버스 계층
  - 버스 경쟁 완화
  - 시스템 버스: 지역 버스, 고속 버스
  - 입출력 버스: 확장 버스, 저속 버스, 장치 버스



[그림 7-9] 버스 계층

7.3 시스템 버스 요약

- 시스템 버스
  - 컴퓨터 구성 요소를 연결하는 신호선의 모임
  - 주소 버스: 기억장치 용량
  - 데이터 버스: 중앙처리장치가 한 번에 처리하는 비트 수
  - 제어 버스: 여러 가지 제어 신호의 모임

7.4 명령어

- 명령어 집합(instruction set)
  - 프로세서가 실행할 수 있는 명령어의 모임
- 학습 목표
  - 명령어 구성 요소와 명령어를 이진수 코드로 표현하는 방법을 이해하고,
  - 컴퓨터가 실행하는 명령어의 종류를 설명할 수 있다.

- 내용

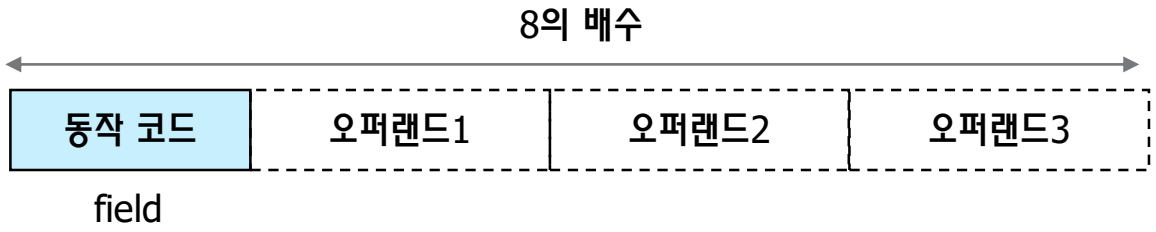
7.4.1 명령어 구성 요소

7.4.2 명령어 종류

7.4.2 오퍼랜드

7.4.1 명령어 구성 요소

- 명령어
  - 하드웨어와 소프트웨어가 만나는 부분
  - 컴퓨터가 수행할 일을 2진수 코드로 표현한 것
- 구성 요소
  - 동작 코드(opcode, operation code)
  - 동작의 대상(operand): 최대 3개
- 명령어 형식
  - 명령어에 포함된 정보를 2진수에 배치한 형태
  - 전체 길이, 필드 구성과 의미, 각 필드의 비트 수

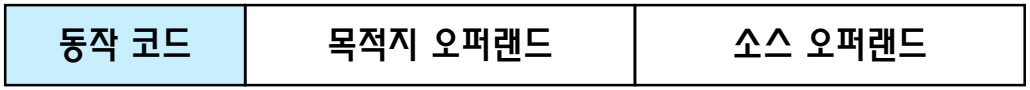


7.4.2 명령어 종류

- 동작 코드에 따른 명령어의 종류
  - 데이터 전달 명령어(data transfer instructions)
    - 내부 레지스터, 주기억장치, 입출력장치 간 데이터 전달 (복사)
  - 데이터 처리 명령어(data process instructions)
    - 내부 레지스터 또는 주기억장치의 데이터를 연산기에서 조작
  - 프로그램 제어 명령어(program control instructions)
    - 프로그램 실행 순서 변경

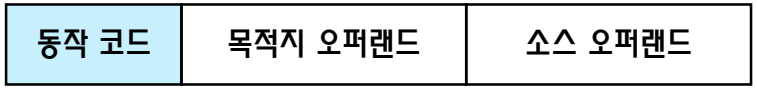
```
int n = 1;              // 데이터 전달
int sum = 0;            // 데이터 전달
while (n <= 100) {      // 프로그램 제어
   sum = sum + n;       // 데이터 처리
   n = n + 1;           // 데이터 처리
}
```

데이터 전달 명령어



- 적재(load): Register ← Mem[address]
- 저장(store): Mem[address] ← Register
- 이동(move): {Register, Mem[address]} ← {Register, Mem[address]}
- 입력(input): Register ← I/O Port[address]
- 출력(output): I/O Port[address] ← Register

데이터 처리 명령어



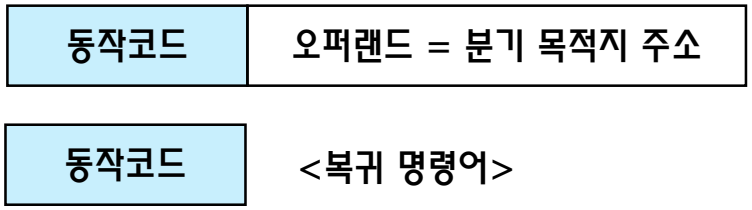
단항 연산 명령어

- 목적지 = op 소스
- Negate (음수, 부호 변경)
- Not, Shift
- 증가, 감소

이항 연산 명령어

- 목적지 = 소스1 op 소스2
- 소스 오퍼런드 한 개 생략 가능:
  목적지 = 목적지 op 소스
- 사칙 연산: +, -, x , /
- 논리 연산: AND, OR, XOR

프로그램 제어 명령어



- 프로그램 실행 순서 변경
  - 궁극적으로 PC의 값 변경

- 프로그램 제어 명령어 종류
  - 무조건 분기: PC ← [분기 주소]
  - 조건 분기: if (조건) PC ← [분기 주소]
  - 서브루틴 호출: PC 저장 / PC ← [서브루틴 시작 주소]
  - 서브루틴 복귀: PC ← [저장된 주소]

[예제]

- [예제 7-3] 어느 그룹의 명령어인가?

1) R1 ← Mem[400] (데이터 전달, 데이터 처리, 프로그램 제어)
2) R2 ← R3 + Mem[401] (데이터 전달, 데이터 처리, 프로그램 제어)
3) PC ← 1004 (데이터 전달, 데이터 처리, 프로그램 제어)
4) Mem[402] ← R2 (데이터 전달, 데이터 처리, 프로그램 제어)
5) if (ZERO) PC ← 1100 (데이터 전달, 데이터 처리, 프로그램 제어)

7.4.3 오퍼랜드

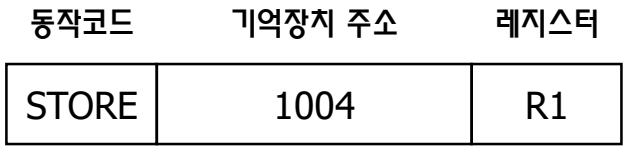
- 오퍼랜드 (operand, 피연산자)
  - 명령어가 처리하려는 동작의 대상
- 유효 데이터 (effective data)
  - 중앙처리장치가 실제로 처리하는 데이터
  - 크기가 정해진 이진수 형태(예: 8 비트, 16 비트, 32 비트의 2진수)
- 오퍼런드 유형
  - 즉치 데이터(immediate data): 명령어가 유효 데이터를 포함한다.
  - 레지스터 이름: 레지스터의 값이 유효 데이터.
  - 기억장치 주소: 기억장치 해당 주소의 값이 유효 데이터.
  - 입출력 포트: 입출력 포트의 값이 유효 데이터.

명령어 형식의 예

(1) R3 ← R3 + 1

(2) R3 ← R3 + 5

(3) Mem[1004] ← R1



(4) R1 ← Port[1004]

(5) R3 ← R1 + Mem[1004]

유효 데이터 해석

- 해석 방법
  - 부호 없는 수 (unsigned number)
  - 정수 (signed number)
  - 실수 (floating-point number)
  - 문자 코드

- 명령어와 오퍼랜드의 관계
  - 명령어에 따라 유효 데이터를 해석하는 방법이 다르다.
  - 데이터의 길이, 해석 방법이 다르면 명령어의 동작 코드도 달라야 한다.

7.4 명령어 요약

- 명령어 집합
  - 프로세서가 실행할 수 있는 명령어의 모임
  - 하드웨어와 소프트웨어 연결
- 명령어 구성
  - 동작코드: 실행할 동작
  - 오퍼랜드: 동작의 대상
- 명령어 종류
  - 데이터 전달 명령어: 복사 전달
  - 데이터 처리 명령어: 연산기에서 계산
  - 프로그램 제어 명령어: 프로그램 실행 순서 변경

7.5 요약

- 7.1 프로그램의 실행
  - 내장형 컴퓨터
  - 명령어 사이클: 인출 단계 + 실행 단계
- 7.2 컴퓨터의 구성 요소
  - 중앙처리장치: 제어장치, 처리장치(레지스터, 연산기)
  - 기억장치: 주소, 데이터, 읽기/쓰기
  - 입출력장치: 포트
- 7.3 시스템 버스
  - 데이터 버스, 주소 버스, 제어 버스(제어신호 종류)
  - 단일 버스, 계층적 버스
- 7.4 명령어
  - 종류: 데이터 전달, 데이터 처리, 프로그램 제어 명령어
  - 구조: 동작 코드 + 오퍼랜드