Recap

- Programmer must keep track of format
- Precision, decimal digits, basis, ASCII
- Unsigned, signed (2's complement)
- Big and little endian

Overview

- Architecture, registers
- Addressing Modes
- Memory Allocation

Architecture: how pieces are connected together

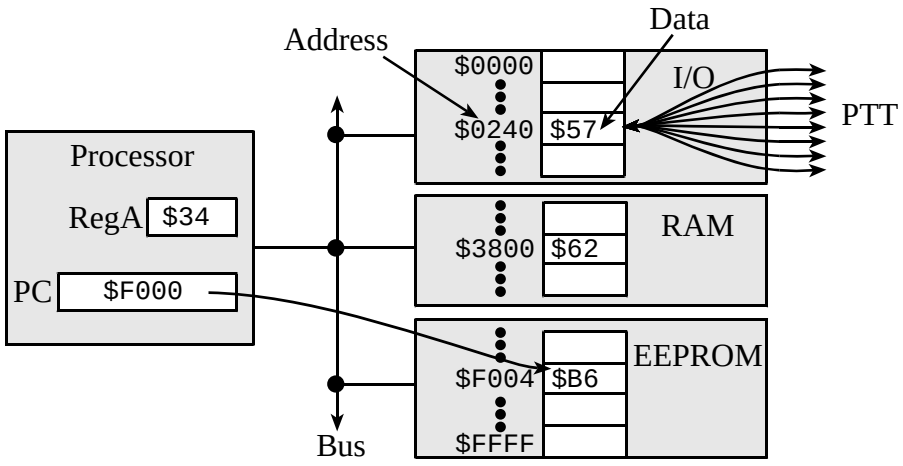


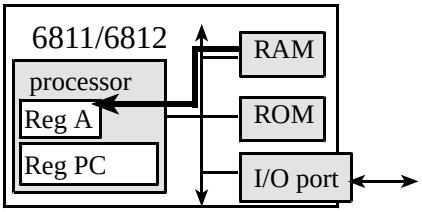
Figure 2.3. The memory model of a simplified 9S12 computer.

To clarify operations that read/write memory,

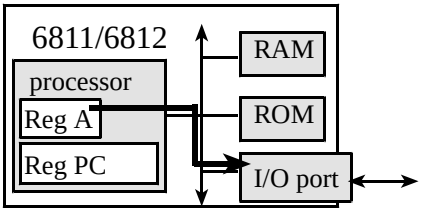
= `[U]` specifies an 8-bit read from address U

`={U}` specifies a 16-bit read from addresses U, U+1
(most significant byte first)
`[U]=` specifies an 8-bit write to address U
`{U}=` specifies a 16-bit write to address addresses U, U+1
(most significant byte first)

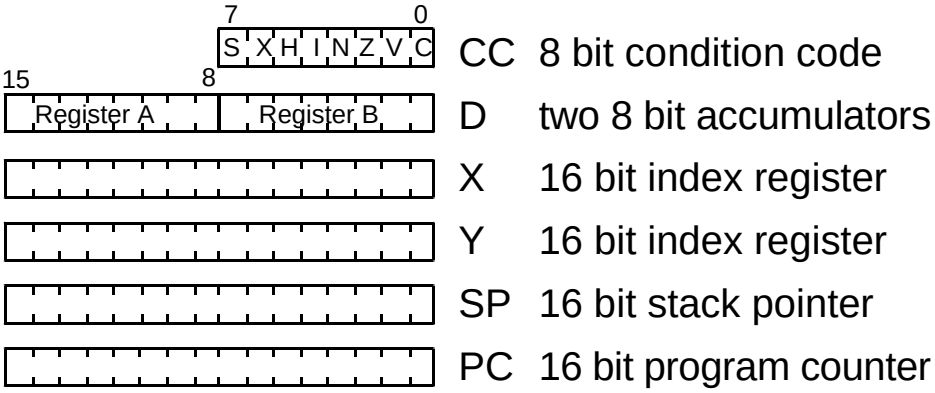
```
ldaa   #w  ;RegA=w
ldaa   U   ;RegA=[U]
staa   U   ;[U]=RegA
bra    U   ;PC=U
```



The `ldaa $3800` instruction reads data from memory



The `staa PTT` instruction stores



2.4. Simplified 9S12 Machine Language Execution

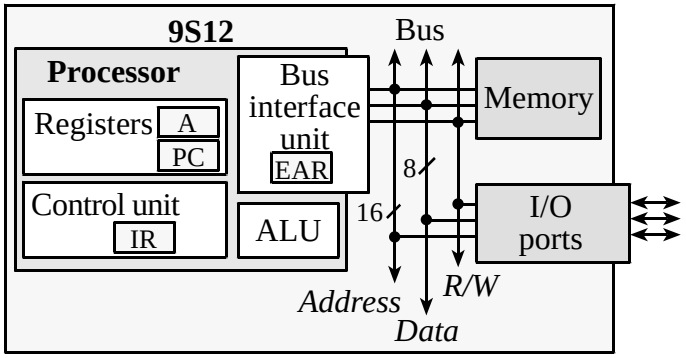


Figure 2.6. Block diagram of a simplified 9S12 computer.

Phase	Function	R/W	Address	Comment
1	**Op code fetch**	**read**	**PC++**	**Put op code into IR**
	Operand fetch	**read**	**PC++**	**Immediate or calculate EA**
2	Decode instruction	none		Figure out what to do
3	Evaluation address	none		Determine EAR
4	**Data read**	**read**	**SP,EAR**	**Data passes through ALU,**
5	Free cycle	read	PC/SP/$FFFF	ALU operations, set CCR
6	**Data store**	**write**	**SP,EAR**	**Results stored in memory**

```
      org   $F000
main  clra          ;RegA=0          inherent mode
      ldaa #36      ;RegA=36         immediate mode
      ldaa $32      ;RegA=[$0032]    direct page mode
      ldaa $3800    ;RegA=[$3800]    extended mode
      bra  main     ;PC=$F000        PC-relative mode
      org  $FFFE
      fdb  main
```

2.5.1. Inherent addressing mode

Action: Start TExaS, new UC (see PC, A), new RTF
Copy paste program, assemble

Observe: See listing file, compare to Figure 2.7

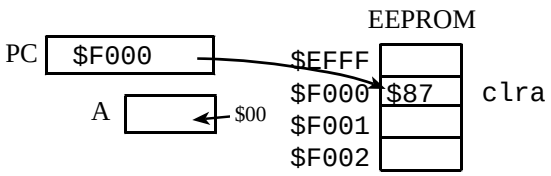


Figure 2.7. Example of the inherent addressing mode (before execution).

Action: Single step

```
Opcode fetch   R 0xF000 0x87 from EEPROM    Phase 1
```

2.5.2. Immediate addressing mode

Observe: See listing file, compare to Figure 2.8

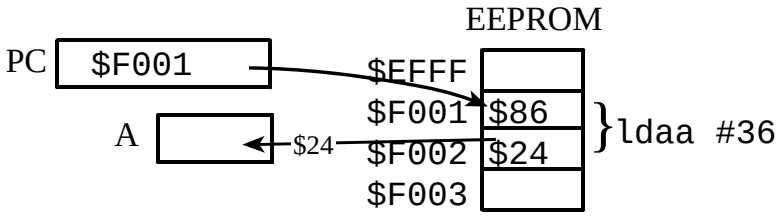


Figure 2.8. Example of the immediate addressing mode (before execution).

Action: Single step

```
Opcode fetch   R 0xF001 0x86 from EEPROM    Phase 1
Operand fetch  R 0xF002 0x24 from EEPROM    Phase 1
```

2.5.3. Direct addressing mode

Observe: See listing file, compare to Figure 2.9

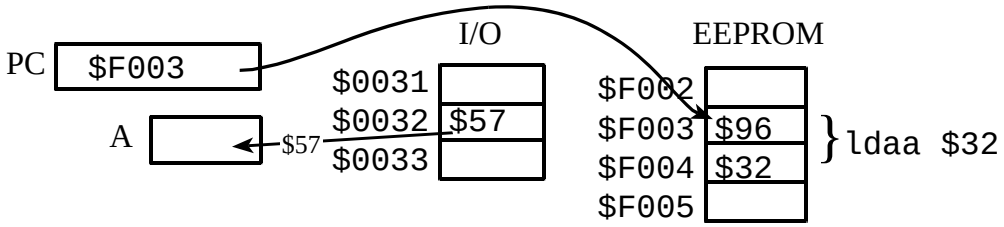


Figure 2.9. Example of the direct-page addressing mode (before execution).

Action: Single step

```
Opcode fetch   R 0xF003 0x96 from EEPROM    Phase 1
Operand fetch  R 0xF004 0x32 from EEPROM    Phase 1
Fetch using EARR 0x0032 0x57 from I/O       Phase 4
```

2.5.4. Extended addressing mode

Observe: See listing file, compare to Figure 2.10

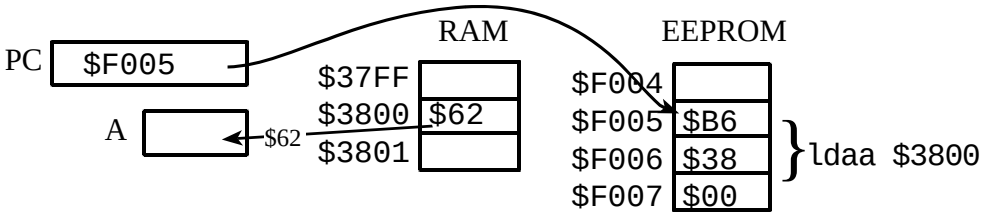


Figure 2.10. Example of the extended addressing mode (before execution).

```
Action:  Single step
Opcode fetch   R 0xF005 0xB6 from EEPROM    Phase 1
Operand fetch  R 0xF006 0x38 from EEPROM    Phase 1
Operand fetch  R 0xF007 0x00 from EEPROM    Phase 1
Fetch using EARR 0x3800 0x62 from RAM       Phase 4
```

2.5.5. PC relative addressing mode

```
Observe: See listing file, look at bra main
Action:  Single step
Opcode fetch   R 0xF008 0x20 from EEPROM    Phase 1
Operand fetch  R 0xF009 0xF6 from EEPROM    Phase 1
```

The bottom line

Computers execute one instruction at a time

- **-Fetch opcode and all operands, PC++ each time**
- **-Decide what to do (decode)**
- **-Determine EAR**
- **-Read data from memory if needed**
- **-Perform operation as needed, sets CCR**
- **-Write data to memory if needed**

Addressing mode specifies how data is obtained

- **-How to calculate EAR**
  - **Inherent (data is not needed or implied)**
  - **Immediate (data in the operand)**
  - **Direct (8-bit address into memory)**

Extended (16-bit address into memory)
PC relative (PC=PC+rr)