표 1

Opcode [24:21]	Mnemonic	Meaning	Effect
0000	AND	Logical bit-wise AND	Rd := Rn AND Op2
0001	EOR	Logical bit-wise exclusive OR	Rd := Rn EOR Op2
0010	SUB	Subtract	Rd := Rn - Op2
0011	RSB	Reverse subtract	Rd := Op2 - Rn
0100	ADD	Add	Rd := Rn + Op2
0101	ADC	Add with carry	Rd := Rn + Op2 + C
0110	SBC	Subtract with carry	Rd := Rn - Op2 + C - 1
0111	RSC	Reverse subtract with carry	Rd := Op2 - Rn + C - 1
1000	TST	Test	Scc on Rn AND Op2
1001	TEQ	Test equivalence	Scc on Rn EOR Op2
1010	CMP	Compare	Scc on Rn - Op2
1011	CMN	Compare negated	Scc on Rn + Op2
1100	ORR	Logical bit-wise OR	Rd := Rn OR Op2
1101	MOV	Move	Rd := Op2
1110	BIC	Bit clear	Rd := Rn AND NOT Op2
1111	MVN	Move negated	Rd := NOT Op2

표 2

값	의미	값	의미
0	EQ (EQual)	8	HI (unsigned HIgher)
1	NE (Not Equal)	9	LS (unsigned Lowe or Same)
2	HS (unsigned Higher or Same)	10	GE (signed Greater than or Equal)
3	LO (unsigned LOwer)	11	LT (signed Less Than)
4	MI (MInus, <0)	12	GT (signed Greater Than)
5	PL (PLus, >=0)	13	LE (signed Less Than or Equal)
6	VS (oVerflow Set, overflow)	14	AL (ALways)
7	VC (oVerflow Clear, no overflow)	15	NV (reserved)

그림 1

```
1           PRESERVE8
2       AREA bubbleSort, CODE, READONLY
3
4       IMPORT    swap
5       EXPORT bubblesort
6  bubblesort
7                   SUB             SP, SP, #20
8                   STR             LR, [SP, #16]
9                   STR             R7, [SP, #12]
10                  STR             R6, [SP, #8]
11                  STR             R3, [SP, #4]
12                  STR             R2, [SP, #0]
13
14                  MOV             R6, R0  ; copy parameter v
15                  MOV             R7, R1  ; copy parameter n
16                  MOV             R2, #0  ; i=0
17
18 for1tst
19                  CMP             R2, R1  ; if i >= n
20                  BGE             exit1   ; go to exit1 if i >= n
21                  SUB             R3, R2, #1      ; j=i-1
22
23 for2tst
24                  CMP             R3, #0          ; IF j < 0
25                  BLT             exit2           ; go to exit2 if j < 0
26                  ???
27                  ???
28                  ???
29                  ???
30                  ???
31
32                  STMDB           SP!, {R0,R1,R2,R3,R12}
33                  MOV             R0, R6          ; first swap parameter is v
34                  MOV             R1, R3          ; second swap parameter is j
35                  BL              swap            ; call swap code
36                  LDMIA           SP!, {R0,R1,R2,R3,R12}
37
38                  SUB             R3, R3, #1      ; j = j -1
39                  B               for2tst
40
41 exit2
42                  ADD             R2, R2, #1      ; i = i +1
43                  B               for1tst
44
45 exit1
46                  LDR             R2, [SP, #0]
47                  LDR             R3, [SP, #4]
48                  LDR             R6, [SP, #8]
49                  LDR             R7, [SP, #12]
50                  LDR             LR, [SP, #16]
51                  ADD             SP, SP, #20
52
53                  MOV             PC, LR
54                  END
```

```
void bubblesort (int v[], int n)
{
  int i, j;
  for (i = 0; i < n; i ++)
    for (j = i - 1; j >= 0 ; j --)
      if (v[j] > v[j + 1])swap(v,j);
}
```

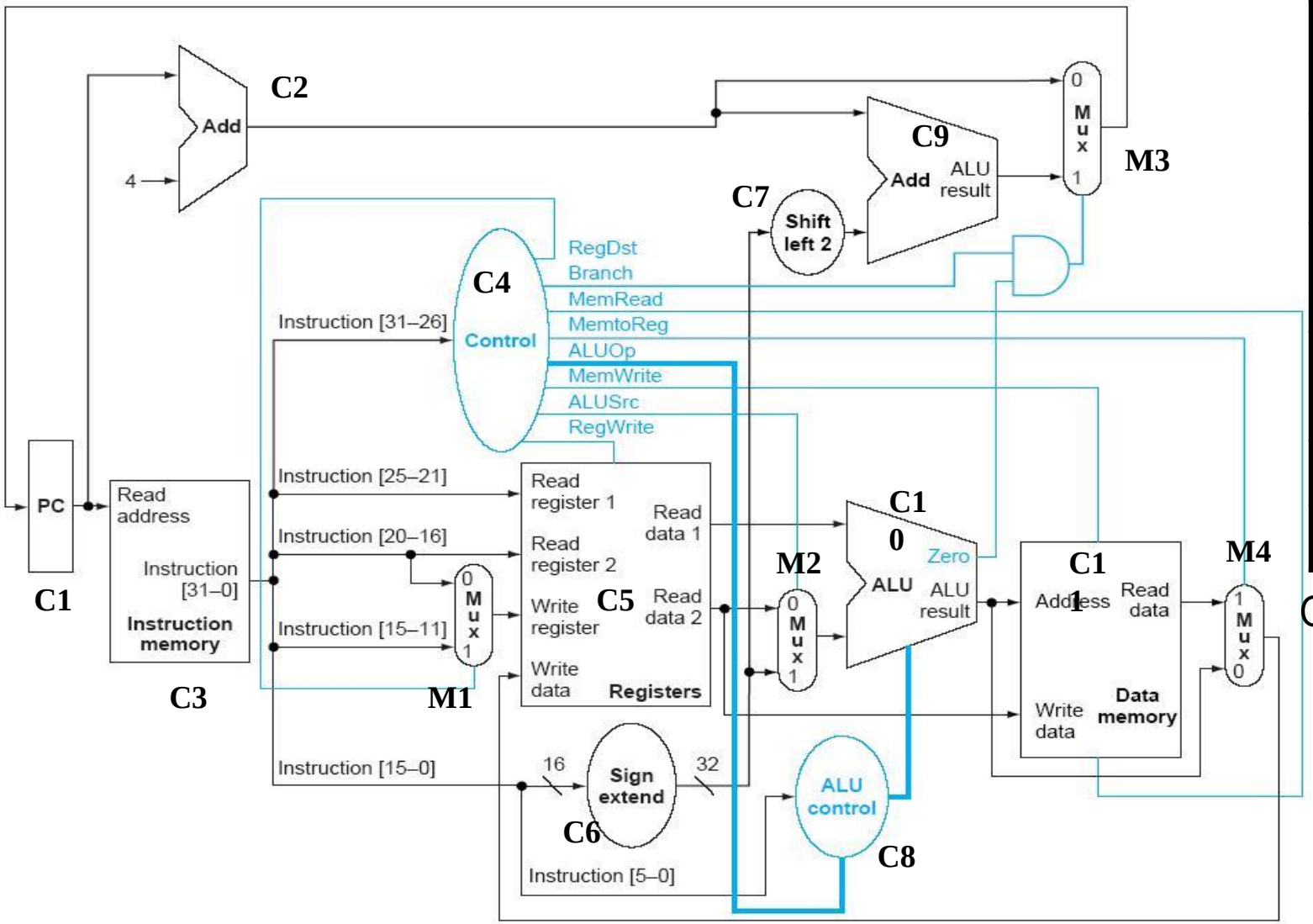
```
1           PRESERVE8
2       AREA Swap, CODE, READONLY
3
4       EXPORT swap
5  swap
6                   ADD             R12, R0, R1, LSL #2
7
8                   LDR             R2, [R12, #0]
9                   LDR             R3, [R12, #4]
10
11                  STR             R3, [R12, #0]
12                  STR             R2, [R12, #4]
13
14                  MOV             PC, LR
15                  END
```

그림 2

```
1         PRESERVE8
2         AREA Ex3, CODE, READONLY
3
4         EXPORT  example3
5
6 example3
7                 STMFD   sp!,{R4-R9,lr}
8                 MOV     R4, R2
9                 MOV     R6, R3
10                MOV     R7, #0
11                MOV     R8, #0
12
13 Loop2
14                MOV     R3, #0
15                MOV     R9, R0
16
17 Loop
18                LDRB    R5, [R9], #1
19                CMP     R4, R5
20                ADDEQ   R3, R3, #1
21                CMP     R9, R1
22                BLS     Loop
23                CMP     R3, R7
24                MOVGT   R7, R3
25                MOVGT   R8, R4
26                CMP     R4, #0XFF
27                ADDLT   R4, R4, #1
28                BLT     Loop2
29
30                STR     R8, [R6]
31                STR     R7, [R6,#4]
32                LDMFD   sp!,{R4-R9,lr}
33
34                MOV     PC, lr
35                END
36
```

```
37 int main() {
38          int result[2];
39
40          example3(0x00,0x100, 0x0, result);
41          sendstr("Top 1 pattern : ");
42          printDecimal(result[0]);
43          sendstr("\n");
44
45          sendstr("Top 1 pattern count : ");
46          printDecimal(result[1]);
47          sendstr("\n");
48 }
```

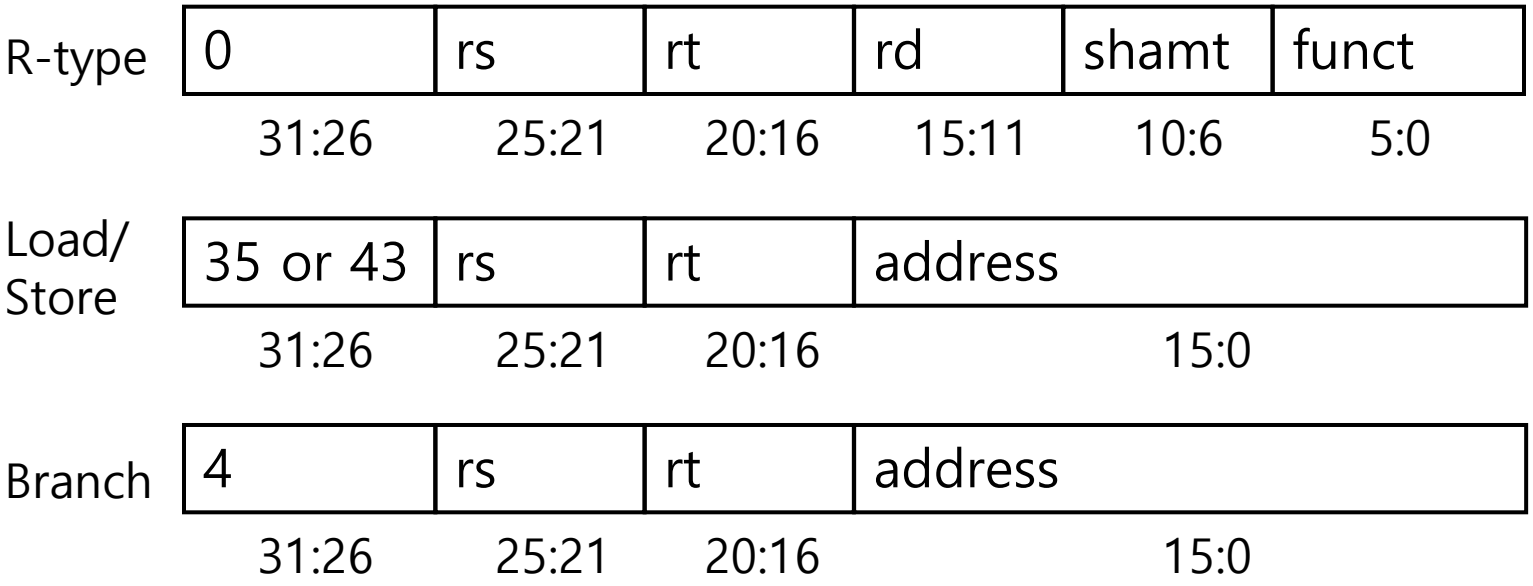
그림 3



RegDst(M1)	
ALUSrc(M2)	
MemtoReg (M4)	
RegWrite	
MemRead	
MemWrite	
Branch	
ALUOp	

Control signal 표

그림 4


R-type
0
rs
rt
rd
shamt
funct
31:26
25:21
20:16
15:11
10:6
5:0
Load/
Store
35 or 43
rs
rt
address
31:26
25:21
20:16
15:0
Branch
4
rs
rt
address
31:26
25:21
20:16
15:0