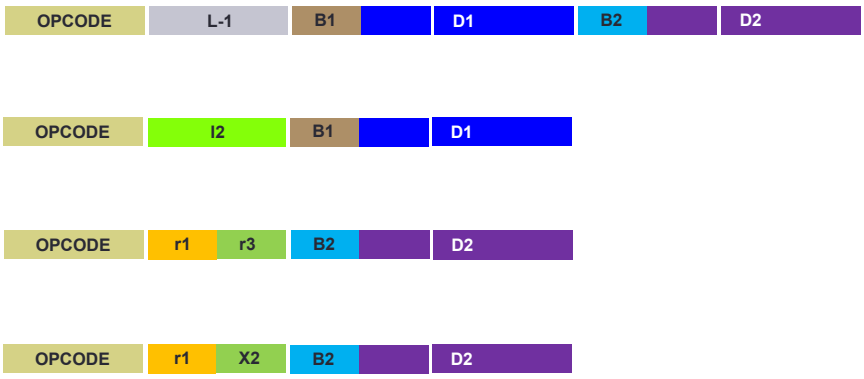
Computer Structure and Language

Hamid Sarbazi-Azad
Department of Computer Engineering
Sharif University of Technology (SUT)
Tehran, Iran

Advanced Instructions

Now, we introduce some advanced instructions in SS1, SI, RX and RS formats.



Advanced Instructions (SS1 Format):

Translate Character

Mnemonic: TR S1(L),S2
TR D1(L,B1),S2
TR S1(L),D2(B2)
TR D1(L,B1),D2(B2)

Operation: for i=0 to L-1 do $M_{D1+(B1)+i} \leftarrow (M_{D2+(B2)+(M\_D1+(B1)+i)})$;

OPCODE: DCh

Note: All SS1 instructions (including TR) work from left to right (i.e. lower address to higher address)

Advanced Instructions (SS1 Format):

Translate Character

Examples:

TR VAR3(1),VAR5+5

Main Memory

0000FCh	--	--	--	F7
000100h	F1	F9	F1	F2
000104h	F3	F4	04	C0
000108h	B1	04	C0	B1
00010Ch	C8	C1	D4	C8
000110h	C1	D4	F3	F3
000114h	F3	F3	F3	5C
000118h	5C	5C	F0	F1
00011Ch	F2	F3	F4	F5
000120h	F6	F7	F8	F9

Symbol Table

Symbol	Address
VAR1	0000FFh
VAR2	000102h
VAR3	000106h
VAR4	00010Ch
VAR5	00011Ah

Advanced Instructions (SS1 Format):

Translate Character

Examples:

TR VAR3(1),VAR5+5 after execution

Assume VAR3 and VAR5 are addressed by base register R12 and (R12) = 6.

Machine code:

TR VAR3(1),VAR5+5

DC00C100C119

Main Memory

0000FCh	--	--	--	F7
000100h	F1	F9	F1	F2
000104h	F3	F4	F9	C0
000108h	B1	04	C0	B1
00010Ch	C8	C1	D4	C8
000110h	C1	D4	F3	F3
000114h	F3	F3	F3	5C
000118h	5C	5C	F0	F1
00011Ch	F2	F3	F4	F5
000120h	F6	F7	F8	F9

Symbol Table

Symbol	Address
VAR1	0000FFh
VAR2	000102h
VAR3	000106h
VAR4	00010Ch
VAR5	00011Ah

Advanced Instructions (SS1 Format):

Translate Character

Example 1: Suppose (M_A) = 0901070300, A = (R12)+45, and Literal address = (R12)+200

Assembly instruction: TR A(5),=C'0123456789'

Operation: $M_A \leftarrow$ F9F1F7F3F0; = C'91730'

Machine code: DC04C02DC0C8

Example 2: Suppose (M_A) = 0901070300, A = (R12)+45, and Literal address = (R12)+200

LA 2,A

Assembly instruction: TR 1(3,2),202(12)

Operation: $M_A \leftarrow$ 09F3F9F500;

Machine code: DC022001C0CA

Advanced Instructions (SI Format):

Test Under Mask

Mnemonic: TM S1,M2
TM D1(B1),M2

Operation: Test the indicated bits by M2 in byte ($M_{D1+(B1)}$) and update CC accordingly:
if all tested bits are 0 then CC ← 00;
if all tested bits are 1 then CC ← 11;
if all tested bits are mixed then CC ← 01;

OPCODE: 91h

Advanced Instructions (SI Format):

Test Under Mask

Example 1:

Assembly instruction: TM ARR,135 ARR = (R12)+100 and (M_{ARR}) = 100

Operation: CC ← 01;

Mask = 10000111

(M_{ARR}) = 01100100

Machine code: 9187C064

Advanced Instructions (SI Format):

Test Under Mask

Example 1:

Assembly instruction: TM ARR,B'11' ARR = (R12)+100 and (M_{ARR}) = 100

Operation: CC ← 00;

Mask = 00000011

(M_{ARR}) = 01100100

Machine code: 9103C064

Advanced Instructions (SI Format):

Test Under Mask

Example 1:

Assembly instruction: TM ARR,36 ARR = (R12)+100 and (M_{ARR}) = 100

Operation: CC ← 11;

Mask = 00100100

(M_{ARR}) = 01100100

Machine code: 9124C064

Advanced Instructions (RS Format):

Insert Character Under Mask

Mnemonic: ICM r1,M3,S2
ICM r1,M3,D2(B2)

Operation: Insert characters from memory at address S2 into r1 according to the mask M3 = $m_3m_2m_1m_0$.
m_3 corresponds to byte $r1_{31..24}$
m_2 corresponds to byte $r1_{23..16}$
m_1 corresponds to byte $r1_{15..8}$
m_0 corresponds to byte $r1_{7..0}$

OPCODE: BFh

Advanced Instructions (RS Format):

Insert Character Under Mask

Example 1:

Assembly instruction: ICM 2,5,ARR ARR = (R12)+100 and
(M_{ARR}) = 12345678h and
(R2) = -1;

Operation: R2 ← FF12FF34h;

Mask = 0101

(R2) = FF FF FF FFh

after execution (M_{ARR}) = 12 34 56 78h
(R2) = FF 12 FF 34

Machine code: BF25C064

Advanced Instructions (RS Format):

Store Character Under Mask

Mnemonic: STCM r1,M3,S2
STCM r1,M3,D2(B2)

Operation: Store bytes of r1 according to mask M3 = $m_3m_2m_1m_0$ into memory bytes at address S2.
m_3 corresponds to byte $r1_{31..24}$
m_2 corresponds to byte $r1_{23..16}$
m_1 corresponds to byte $r1_{15..8}$
m_0 corresponds to byte $r1_{7..0}$

OPCODE: BEh

Advanced Instructions (RS Format):

Store Character Under Mask

Example 1:

Assembly instruction: STCM 3,3,ARR ARR = (R12)+100 and
(M_{ARR}) = 12345678h and
(R3) = 0

Operation: (M_{ARR}) ← 00005678h;

Mask = 0011

(R3) = 00 00 00 00h

after execution

(M_{ARR}) = 00 00 56 78

Machine code: BE33C064

Advanced Instructions (RS Format):

Compare Logical Character Under Mask

Mnemonic: CLM r1,M3,S2
CLM r1,M3,D2(B2)

Operation: Compare byte string in r1 according to mask M3 = $m_3m_2m_1m_0$ with memory bytes stored at address S2 and update CC; Mask bits M3 are decoded as:
m_3 corresponds to byte $r1_{31..24}$
m_2 corresponds to byte $r1_{23..16}$
m_1 corresponds to byte $r1_{15..8}$
m_0 corresponds to byte $r1_{7..0}$

OPCODE: BDh

Advanced Instructions (RS Format):

Compare Logical Character Under Mask

Example 1:

Assembly instruction: CLM 3,9,ARR ARR = (R12)+100 and
(M_{ARR}) = 12345678h and
(R3) = 12009834

Operation: CC ← 00;

Mask = 1001

(R3) = 12 00 98 34h
(M_{ARR}) = 12 34 56 78

Machine code: BD39C064

Example 1: Write an assembly program to generate all subsets of set "ABCD".

```
SUBSTR START  0
[ Defining R12 as base register
[ & initialize it to 6  → (R12) =6.
         LA     2,15
         LA     3,SUBS
         ICM    4,15,STR
LOOP     LA     5,64(2)
         STC    5,LAB+1
LAB      STCM   4,0,0(3)
         LA     3,4(3)
         BCT    2,LOOP
[ Returning to OS
STR      DC     C'ABCD'
SUBS     DC     16CL4' '
         END    SUBSTR
```


Example 1: Write an assembly program to generate all possible substrings of string "ABCD".

Address	Code	Assembly Instruction		
000000		SUBSTR	START	0
		[Defining R12 as base register		
		[& initialize it to 6 → (R12) =6.		
000006	4120000F		LA	2,15
00000A	4130C03A		LA	3,SUBS
00000E	BF4FC036		ICM	4,15,STR
000012	41520040	LOOP	LA	5,64(2)
000016	4250C015		STC	5,LAB+1
00001A	BE403000	LAB	STCM	4,0,0(3)
00002E	41330004		LA	3,4(3)
000032	4620C00C		BCT	2,LOOP
000036		[Returning to OS		
00003C	C1C2C3C4	STR	DC	C'ABCD'
000040	40404040	SUBS	DC	16CL4' '
			END	SUBSTR

Symbol Table

Symbol	B	Disp.
LOOP	C	00Ch
LAB	C	014h
STR	C	036h
SUBS	C	03Ah

Advanced Instructions (RX Format):

Execute Instruction

Mnemonic: EX r1,S2(X2)
EX r1,S2
EX r1,D2(X2,B2)
EX r1,D2(X2)
EX r1,D2(,B2)

Operation: Copy the instruction at S2+(X2) into a temp register T;
OR the second byte of T with byte $(r1)_{7..0}$;
Execute the instruction in T;

OPCODE: 44h

Note: After executing the instruction in T the execution continue by the instruction located after EX instruction in the program, unless the instruction in T is a branch or a call instruction.

Example 2: Write an assembly program to add some decimal numbers P_1, P_2, P_n, of different lengths stored in array A. The length of element P_i is stored in element L_i of a byte array L (L_{n+1}=Null). Store the result in word SUM.

```
ACCUM START  0
[ Defining R12 as base register
& initialize it to 6  → (R12) =6. ]
         LA     2,L
         LA     3,A
         ZAP    JAM(8),=P'0'
LOOP     CLI    0(2),0
         BE     FIN
         IC     4,0(2)
         BCTR   4,0
         EX     4,ADD
         LA     2,1(2)
         LA     3,1(3,4)
         B      LOOP
FIN      CVB    2,JAM
         ST     2,SUM
```

Homework:
Generate the machine code of this program.

```
[ Returning to OS ]
ADD      AP     JAM(8),0(0,3)
JAM      DS     D
L        DC     X'0201….03',X'0'
A        DC     PL2'+23',PL1'-3',…, PL3'234'
SUM      DS     F
         END    ACCUM
```

End of Slides