2016 Spring

(2lb)

Line	Location	Symbol	opcode	Exp	Objectcode
10	0000	TEST	START	0	—
20	0000	FIRST	LDA	#3	010003
30	0003		STX	THREE	132013
40	0006		LDX	=X'00'	072010
50	0009		+LDS	THREE	6F100016
60	000D		ADDR	A,X	9001
70	000F		+STA	RESULT,X	0F90013
80	0013	RESULT	RESW	1	—
90	0016	THREE	RESW	1	—
100	0019		END	FIRST	—
110	0019	*	=X'00'		00

start from Here

[redacted]:

Location fill गर्ने बेलामा opcode मात्र ध्यान दिने

(i) Directive (START, END, BASE) हरुले address occupy Gardaina

(ii) यदि opcode मा
Suru मा + (+LDS) or Last मा R
(ADDR) आउछ भने its SIC/XE

(iii) SIC simple मा सबै instruction 3 Byte को हुन्छ But ~~4 Byte~~ SIC/XE मा 1, 2, 3 or 4 byte instruction format नि हुन्छ।

(iv) यदि operand like having '+' in prefix (सुरुमा) then its 4 byte (+LDS)
यदि operand like having 'R' in suffix (Last) मा then its 2 byte
यदि केहि पनि छैन भने its 3 byte.

Line 10 : its operand: START

यो START directive को operand ले पनि address specify garekO हुन्छ त्यहाँ बाट address start हुन्छ 'Location' column मा

START is direction it doesnot take any address '0000'

Line 20:

START ले कति पनि ~~[illegible]~~ address लिएन so line 20 start from 0000

But its operand LDA जसमा '+' नि हैन 'R' नि हैन so its 3 byte ~~Add~~ it occupy

|0000/0001/0002/

$\underbrace{\hphantom{0000/0001/0002}}_{LDA \ \#3}$

Line 30

Line 20 ले 0000, 0001, 0002, occupy गर्यो So Line 30 starts froom 0003

यो पनि (STX) 3 byte it occupy

|0003|0004|0005|

Line 40

start from 0006 and occupy 3 byte upto 0008

0006|0007|0008

Line 50

(+) LDS

→ 4 byte

~~|0008|0009|~~000A|000B| ✗

|0009|000A|000B|000C| ✓

Line 60

ADD (2) → 2 byte

| 000D | 000E |

Line 70

+STA

000F | 0010 | 0011 | 0012

Line 80

RESW ocupy specified word (1 word = 3 Byte)
so RESW (1) → it occupy 1 word ie 3 Byte
[it occupy 3 byte.]

S U P P O S E

If it was RESW (17) → it occupy 17 ओटा word (in decimal) which is 17×3 = 51 Byte (in decimal) so it will occupy 51 address and translate into hex

Suppose

16 | 51 3 = 33
3

[so Add 33 in address (because our address is in hex)

start from 0013 and occupy 3 Byte
0013 | 0014 | 0015

Line 90 :

Same case: 0016 / 0017 / 0018 /

Line 100 :

Line 90 ले 0018 सम्म लिन्छ, अनि 0019 बाट 100 execute. and its a directive it doesnot ocupy any memory.

Line 110 :

for Line 40 =' x '00' → Literal (यसको लागि pool बनाको।

* → current address

For object code

Object code को लागि expression हेर्ने
Target Address = operand को address
Like Line 30: STX THREE, THREE को address हो (in Line 90) ∴ T.A = 0016

So basically object code of SIC/XE compose ~~of~~ on (instruction format)

In 2 Byte:

opcode	R1	R2
8 bit	4 bit	4 bit

In 3 Byte

opcode	n	i	x	b	p	e	displacement
6 bit	← 2 →		← 4 →				← 12 →

← 24 bit → (3 bit)

In SIC simple:

Object code = ~~operand~~ opcode 'concat' with target address

In 4 Byte

opcode	n	i	x	b	pe	displacement
← 8 →	← 4 →					← 20 bit →

IDEA:

n i को value select गर्ने तरिका
यदि exp (operand) मा
@ → add 2 in opcode
# → add 1 in opcode
कोहि पनि छैन भने add 3 in opcode.
} Rule 1

displacement = (TA − PC) आफ्नो instruction को मुनि (one step below)

~~A~~ X को value ~~@~~ generally 0 हुन्छ, But some instruction like (line 70) RESULT, X थप्दा मा चाहिं X = 1

b	p	
0	0	→ 4 byte instruction (+ LDS) भएको b=0, p=0
1	0	} 3 byte मा चाहिं suru मा p=1, b=0
0	1	राख्ने, (~~if displa~~ पर्छ भने)

e	
0 →	3 byte instruction को e zero हुन्छ।
1	4 byte instruction को e 1 हुन्छ।

→ Line 10

directive (START) No object code

→ Line 20

LDA #3 # (see rule 1) add 1 in opcode

↳ 00
+1
01

{ # छ भने, direct 3 लाई put गर्ने as a displacement (so 01XXXX)

$\boxed{010003}$ write 3 interms of 4 bit

→ Line 30

STX THREE (see rule 1) add 3

↳ 10
+3
13

x	b	p	e
0	0	1	0

↳ No (,X) $\underbrace{\quad}_{2}$

↳ 3 byte हो, 4 Byte हैन (no +, no R)

```
disp = TA      (see address of THREE
     _ PC      (see line 40's address)
                 ↳ 1 step below's address

disp = 0016
      -0003
      0|013| → Take last 3 digit for
               3 Byte instruction (you
```

Know it is 3 Byte instruction Right, Because NO '+', NO R.

so object code : 132013

Line 40

```
+LDS   =X'00' → its literal because of '='
  ↳ 6C    x  b  p  e  | =X'00' को
    +3    0  0  0  1  | address लाइन मा
    6F          0 1      बनाको छ, 110 मा
C+3 = D+2 = E+1 = F
```

```
disp = TA     0019      |6F2010|
     - PC    -0009
             0|010|     WRONG
```

its 4 Byte NO PC, NO Base
So X=0, b,p = 0,0 and e=1

disp = Target Address of 20 bit (5 ~~bi~~ character)
00019
↳ यो zero add gareko 5 वटा बनाउन

$\boxed{6F100019}$

Line 40

LDX =X'00' → its literal because of '='

↳ 04 +3 = 07

x	b	p	e
0	0	1	0

2

=X'00' का address Last मा छे, 110 मा

disp = TA 0019
− PC 0009
= 0 010

$\boxed{072010}$

072010

Line 50

+LDS THREE

↳ 6C +3 = 6F 1

x	b	p	e
0	0	0	1

its 4 byte so, b & p = 0 and e = 1, x = 0 (x=1 चाहि THREE,X आको आए)

disp = Target address of THREE ie 0016
but its 4 byte, we need 20 byte of displacement [see page 4 (4 byte instruction format]
so make 0016 of 5 bit by adding one 0.
→ 00016

So object code $\boxed{6F100016}$

Line 60 :

ADDR A,X Ⓡ → 2 byte

Opcode ADDR	R1 A	R2 X
90	0	1

↳ 90 40 ↳1 A=0 X=1

A	X	L	B	S	T	F	PC	SW
0	1	2	3	4	5	6	8	9

$\boxed{9001}$

Line 70

+ STA RESULT, X   [it 4 byte b=0, p=0, e=1

$$\begin{array}{ll} \text{OC} & x\ b\ p\ e \\ \underline{+3} & \underline{1\ 0\ 0\ 1} \\ \text{0F} & 9 \end{array}$$

'+' आयो भने 4 byte so,

$$\boxed{\text{displacement} = \text{TA} - \text{PC}}$$ हैन

यो 3 byte लाई मात्र हो

displacement = TA मात्र (but 20 byte)
(Line 50 को जस्तै)

disp = 00013 $\boxed{\text{0F900013}}$

Line 80, 90, 100 को कन्टैन [directive]

Line 110 → conver x'00' into hex
Oh! its already in hex because x' is hex
so object code

$\boxed{00}$

अहिले सम्म (i) Location done
(ii) Object ~~program~~ code done

(iii) Data structure

SYMTAB

Name	value
Test	0000
first	0000
RESULT	0013
THREE	0016

→ Location भनें Table बाट page 1 को

LITTAB

Literal expression	value	length	address
=x'00'	00	1	0019

(iv) Object code file/program

→ H^program name^start^length
↳ program name = Test (See table page 1)
start address = 0000
length = end - start
(0019 + 1) - 0000 [1 chai =x'00' को लिन्छ so add 1]
0020 - 0000
0020

→ T^start address^length^object codes ----
↳ object code को start address
↳ length (too tricky) [see Note/Book]
~~max~~ ↳ object code : list गर्ने
length chai, maxm object code को digit 60 samma include गर्न milxa.

36 ÷ 2 = 18
12

object code

```
H^Test^000000^000020
T^000000^12^010003^132013^072010^
    6E100016^9001^0F9000
T^000019^01^00
M^00000A^05
M^0000ΦΘ^05
E^000000
```