『系統程式』—課程作業(2)

評　分

File Name: E236600-H-AY111A-02.doc

一、注意事項：

(1) 本作業繳交方式：以上傳電子檔方式繳交，並應配合下列之說明。

(A) 請使用本作業電子檔之作答頁依題序完成本作業，於首頁填入個人識別資料後，以[學號]-**HW02.doc** 之格式命名存檔並繳交。

(B) 將完成本作業相關檔案(含本作業檔案)收集於一資料夾內(資料夾之命名方式：[學號]-**HW2**)，並附加資料夾內容說明檔；將該資料夾壓縮後，上傳至成功大學數位學習平台(Moodle)之指定位置。壓縮檔名之格式為：[學號]-HW02.(rar|zip)，壓縮格式可為 rar 或 zip。

(2) 切勿更改本作業原始版面設定，並依規定保存所有相關資料與程式等檔案。

(3) 獨立完成作業，如有抄襲或剽竊情事，除酌情議處並扣減學期成績。

(4) 評分基準：各題配分× 83%，優增劣減。遲交酌減得分，扣減方式另行公佈。

二、基本資料：

繳交期限：**2022/12/06**　　繳交日期：**2022/ 12 / 01**

班級： 電機113　**學號：** F64096114　**姓名：** 郭家佑

三、作業內容：(Main Theme – SIC Assembler and Relocation Loader)

(1) Do the exercise 2 on page 114 of the course textbook。 (10 points)

(2) Do the exercise 8 on page 115 of the course textbook。 (10 points)

(3) Leland L. Beck 提供之 SIC Simulator 其操作指令 E Rx Hex-Value 可設定 SIC CPU 暫存器 x 之值，但不可設定 Program Counter (PC) 之值。試設計一操作程序可於 SIC Simulator 使用中設定暫存器 PC 之值。 (20 points)

(4) 試設計與實作一適用於 SIC/XE 之 Relocation Loader，使用一陣列(Array)模擬 16 KB 記憶體儲存空間，可將如課程用書 pp. 65 Figure 2.8 之 Object Program 載入隨機分配之記憶體儲存空間，並將載入結果產生文字(Text)格式之記憶體映像檔(Memory Image File)以供使用 SIC Simulator 匯入並執行。 (60 points)

- 輸入資料: Object code program for SIC or SIC/XE (example. pp. 49 & pp. 65)
- 輸出資料: ASCII Text File of Memory Image as described below.

Line	Text Content and Format	Note
1	A[char(6)][Hex(6)][Hex(6)][Hex(6)]	Header line: **I**/Name/Saddr/Size/Taddr
2 ~ m	xxxxxxxx … xxxxxxxx … xxxxxxxxx xxxx … xxxx … xxxx … xxxx … xxx xxxxxxxx … xxxxxxxx … xxxxxxxxx	Memory bytes in ASCII Hex Digits ① 32 Bytes/Line (64 chars/Line) ② 2 Hex Digits/Byte
(m+1)	xxxxxxxx	Last line of memory bytes (<= 64 char)

* Notes: (A) File name: **DEVF2** when imported by SIC simulator
(B) For relocatable object program files, the M record described on page 64 in course textbook is followed.

Course Textbook: "System Software – An Introduction to Systems Programming", by Leland L. Beck 3rd Ed.

✍ 請使用本頁起之空間完成作業。(注意事項：請標明題號並依題序完成作業。)

(1)

2. As we have described it, the BASE statement simply gives information to the assembler. The programmer must also write an instruction like LDB to load the correct value into the base register. Could the assembler automatically generate the LDB instruction from the BASE statement? If so, what would be the advantages and disadvantages of doing this?

通常會用到 LDB 指令就會與 base register 一起使用，所以若是 assembler 想自動產生 LDB 指令，programmer 就必須告知它想 load 到 base register 的值。

優點:
因為是 assembler 自動產生指令，所以 programmer 就可以省去花費在要 load 到 base register 的力氣，也大幅減少 programmer 容易漏寫這條指令所需要去 debug 的時間。

缺點:
很明顯地，若是由 assembler 自動產生 LDB 指令，那麼要跳去的 target address 與 programmer 所想要去的會不同，造成程式 error。若想解決這類的問題，programmer 可以在寫 assembler code 時就用 assembler 會自動產生的 base register 的值去計算相對位置，就可避免錯誤。

(2)

8. Our discussion of SIC/XE Format 4 instructions specified that the 20-bit "address" field should contain the actual target address, and that addressing mode bits *b* and *p* should be set to 0. (That is, the instruction should contain a direct address—it should not use base relative or program-counter relative addressing.)

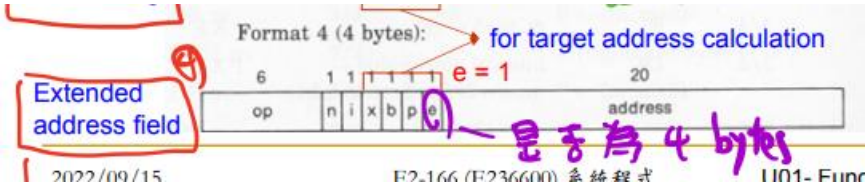
   However, it would be possible to use program-counter relative addressing with Format 4. In that case, the "address" field would actually contain a displacement, and bit *p* would be set to 1. For example, the instruction on line 15 in Fig. 2.6 could be assembled as

```
0006   CLOOP   +JSUB   RDREC          4B30102C
```

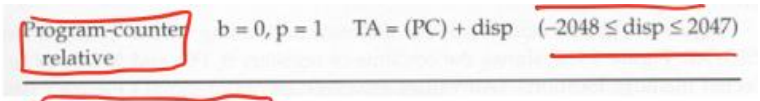
   (using program-counter relative addressing with displacement 102C).

   What would be the advantages (if any) of assembling Format 4 instructions in this way? What would be the disadvantages (if any)? Are there any situations in which it would *not* be possible to assemble a Format 4 instruction using program-counter relative addressing?

Format 4

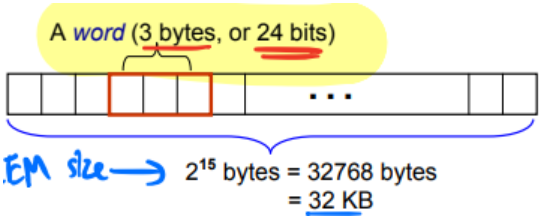


Pc relative/direct addressing

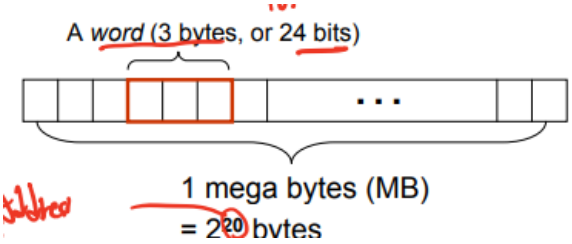


- Direct addressing for formats 3 and 4 if b=p=0

SIC MEM



SIC-XE MEM



優:Pc-relative 與 direct addressing 的差別就在於:Pc-relative 對記憶體位置沒有絕對位置，因此不一定要占用哪一區的記憶體，而 direct 反之。所以若是 direct addressing 遇到要 reference 多個 assembly code，但是遇到兩份以上的 code 要占用同一個記憶體位置就會打架，導致程式 error，而 Pc-relative 就不會發生這總情形。

缺:相比 direct addressing，Pc-relative 每次在算 TA 時都必須做 PC + displacement 的動作，增加硬體的計算量與執行時間。

■ Addressing modes - Determining operand address:

Mode	n	i	x	b	p	e	Target Address (TA)	Note
Immediate	0	1	-	0	0	x	no TA. Value is (12 or 20 bit)*disp*	XE
Register	op		R1				Operands are values in R1 and R2	XE 2
Simple direct	0	0	0				TA = (15 bit) *address* (b-p-e-*disp*)	SIC
Indexed direct	0	0	1				TA = (15 bit) *address* + (X)	SIC
XE simple direct	1	1	0	0	0	0	TA = (12 bit unsigned)*address*	XE 3
XE Direct	1	1	0	0	0	1	TA = (20 bit unsigned)*address*	XE 4
XE Indexed direct	1	1	1	0	0	0	TA = (12 bit unsigned)*disp*+(X)	XE 3
Base relative	1	1	0	1	0	0	TA = (B)+(12 bit unsigned)*disp*	XE 3
PC relative	1	1	0	0	1	0	TA = (PC)+(12 bit signed)*disp*	XE 3
Indexed Base relative	1	1	1	1	0	0	TA = (B)+(12 bit unsigned)*disp*+(X)	XE 3
Indexed PC relative	1	1	1	0	1	0	TA = (B)+(12 bit signed)*disp*+(X)	XE 3
Indirect	1	0	-	x	x	x	Operand address = (TA)	XE

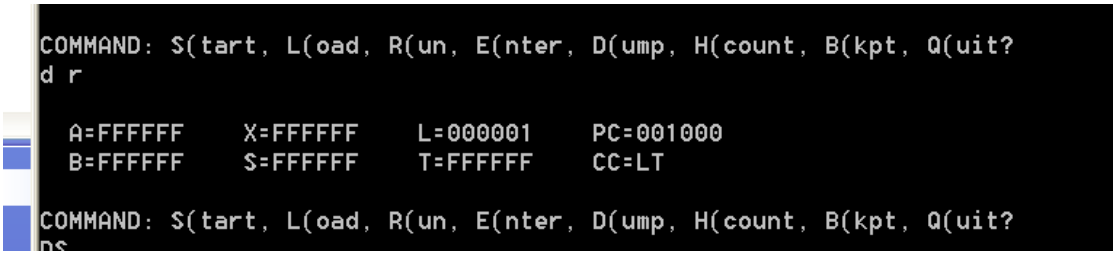
情況:
我認為每個指令都可以用 PC-relative，原因是若是 SIC displacement 有 12bits/SIC-XE 20bits 都剛好符合記憶體大小，所以不會有跳不夠遠的問題，即使 PC 的極端的小/大都可以滿足。

(3)

(3) Leland L. Beck 提供之 SIC Simulator 其操作指令 E Rx Hex-Value 可設定 SIC CPU 暫存器 x 之值，但不可設定 Program Counter (PC) 之值。試設計一操作程序可於 SIC Simulator 使用中設定暫存器 PC 之值。 (20 points)

(sol1)

寫 assembly code 並把 start 的位置設為想要的位置(absolute address)，再把 register dump 出來。



(sol2)

利用 J 指令跳去想要的 PC 位置，並 dump 出來。

(sol3)

利用最暴力的手法就是利用寫 assembly code 來跑 trash instruction，然後 H 設為 1，一直案 R 跑直到想要的 PC 位置出現為止，再把 register dump 出來。

(4)

- 設計 relocation loader 摘要:
  1. 先處理 Hrecord，將 name, start address, program length 存入指定的變數，接著判斷 start address 是否為 000000 來決定要不要 relocate。若是的話，則用 rand 函數產生一個 32 倍數的 offset 給 Loader 並加上 PC 的位置；若不是則直接用他給的絕對位置去做指令。
  2. Trecord 則是直接將後面的資料全部存入一維 vector 內(大小為 program length)，index 是利用該**相對** PC 的位置(需扣掉 offset)。
  3. Mrecord 則是先利用要改 TA 的 length 是奇數還是偶數來做判斷。若是奇數則前面所得到的 byte 的起始位置要*2 再+1；而偶數只用*2。把要改的 TA(string)取出來後再利用寫好的函數把它轉為 int(Dec)並加上 offset 後，再轉回 string(HEX)。
  4. Erecord 則是判斷 transfer address 有沒有跟 start address 一樣，若有的話則進入 output file 的動作。
  5. Output file 則是將 image file 第一行依照指定的格式輸入變數值，後面 2~m 行則是輸出 vector 裡每 64 個的 char(Hex)，輸出完後換行繼續輸出，而因為原本 vector 初始化為 F，所以若是 RESB 或 RESW 的部分會順便被初始化為 0。

- 範例程式:
  - SIC

```
HCOPY  001000001007A
T001000 1E 14 1033 48 2039 00 1036 28 1030 30 1015 48 2061 3C 1003 00 102A 0C 1039 00 102D
T00101E 15 0C 1036 48 2061 08 1033 4C 0000 45 4F 46 000003 000000
T002039 1E 04 1030 00 1030 E0 205D 30 203F D8 205D 28 1030 30 2057 54 90 39 2C 205E 38 203F
T002057 1C 1010 36 4C 0000 F1 001000 04 1030 E0 2079 30 2064 50 9039 DC 2079 2C 1036
T002073 07 3820644C000005
E001000
```

Figure 2.3 Object program corresponding to Fig. 2.2.

  - SIC-XE

```
HCOPY  000000001077
T000000 1D 17 202D 69 202D 4B 1010 36 0320262900003320074B10105D3F2FEC032010
T00001D 13 0F 2016 01 0003 0F 200D 4B 10105D3E2003454F46
T001036 1D B4 10B4 00 B4 40 75101000E32019332FFADB2013A00433200857C003B850
T001053 1D 3B2FEA 13 4000 4F 0000 F1 B4 10774000E32011332FFA53C003DF2008B850
T001070 07 3B2FEF 4F 000005
M000007 05
M000014 05
M000027 05
E000000
```

Figure 2.8 Object program corresponding to Fig. 2.6.

- 執行結果:
  - SIC
    - Loader 後的 image file

```
C:\Documents and Settings\SP User\桌面\simulator>SICSIM-R.exe
SIC SIMULATOR Ver. 1.91 (202111.R02)
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
L
* Loading object code program ...  [ICOPY  001000000107A001000]

  Total Object Code Records: 33

   + Type of object code: Memory image
   + Program/Image NCOPY
   + Starting Memory Address: 001000
   + Size of OBJ Program/Image: 00107A
   + Transfer Address: 001000

* OBJ Code Program/Memory Image Loaded.
```

Start address(absolute): 1000

因此可以 dump 1000-1050 和 2000-2100 的 image file 出來看

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 1000-1050

01000 14103348 20390010 36281030 30101548
01010 20613C10 0300102A 0C103900 102D0C10
01020 36482061 0810334C 0000454F 46000003
01030 000000FF FFFFFFFF FFFFFFFF FFFFFFFF
01040 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
01050 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF

COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
S
```

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 2000-2100

02000 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02010 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02020 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02030 FFFFFFFF FFFFFFFF FF041030 001030E0
02040 205D3020 3FD8205D 28103030 20575490
02050 392C205E 38203F10 10364C00 00F10010
02060 00041030 E0207930 20645090 39DC2079
02070 2C103638 20644C00 0005FFFF FFFFFFFF
02080 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02090 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020A0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020B0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020C0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020D0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020E0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020F0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02100 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF

COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
```

◆ Golden file（用課本範例 object program 跑的 ）

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
l
* Loading object code program ...  [HCOPY  00100000107A      ]

  Total Object Code Records: 7

   + Type of object code: Absolute object code
   + Program/Image NCOPY
   + Starting Memory Address: 001000
   + Size of OBJ Program/Image: 00107A
   + Transfer Address: 001000

* OBJ Code Program/Memory Image Loaded.
```

Start address(absolute): 1000

因此可以 dump 1000-1050 和 2000-2100 的 image file 出來看

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 1000-1050

01000 14103348 20390010 36281030 30101548
01010 20613C10 0300102A 0C103900 102D0C10
01020 36482061 0810334C 0000454F 46000003
01030 000000FF FFFFFFFF FFFFFFFF FFFFFFFF
01040 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
01050 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
```

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 2000-2100

02000 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02010 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02020 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02030 FFFFFFFF FFFFFFFF FF041030 001030E0
02040 205D3020 3FD8205D 28103030 20575490
02050 392C205E 38203F10 10364C00 00F10010
02060 00041030 E0207930 20645090 39DC2079
02070 2C103638 20644C00 0005FFFF FFFFFFFF
02080 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02090 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020A0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020B0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020C0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020D0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020E0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
020F0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02100 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF

COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
```

- SIC-XE
  - Loader 後的 image file

```
C:\Documents and Settings\SP User\桌面\simulator>SICSIM-R.exe
SIC SIMULATOR Ver. 1.91 (202111.R02)
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
L
* Loading object code program ...  [ICOPY  0003000001077000300]

  Total Object Code Records: 33

   + Type of object code: Memory image
   + Program/Image NCOPY
   + Starting Memory Address: 000300
   + Size of OBJ Program/Image: 001077
   + Transfer Address: 000300

* OBJ Code Program/Memory Image Loaded.
```

Start address(relocatable): 000300

因此可以 dump 000300-000350 和 001300-001400 的 image file 出來看

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 300-350

00300 17202D69 202D4B10 13360320 26290000
00310 3320074B 10135D3F 2FEC0320 100F2016
00320 0100030F 200D4B10 135D3E20 03454F46
00330 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
00340 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
00350 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
```

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 1300-1400

01300 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
01310 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
01320 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
01330 FFFFFFFF FFFFB410 B400B440 75101000
01340 E3201933 2FFADB20 13A00433 200857C0
01350 03B8503B 2FEA1340 004F0000 F1B41077
01360 4000E320 11332FFA 53C003DF 2008B850
01370 3B2FEF4F 000005FF FFFFFFFF FFFFFFFF
01380 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
01390 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
```

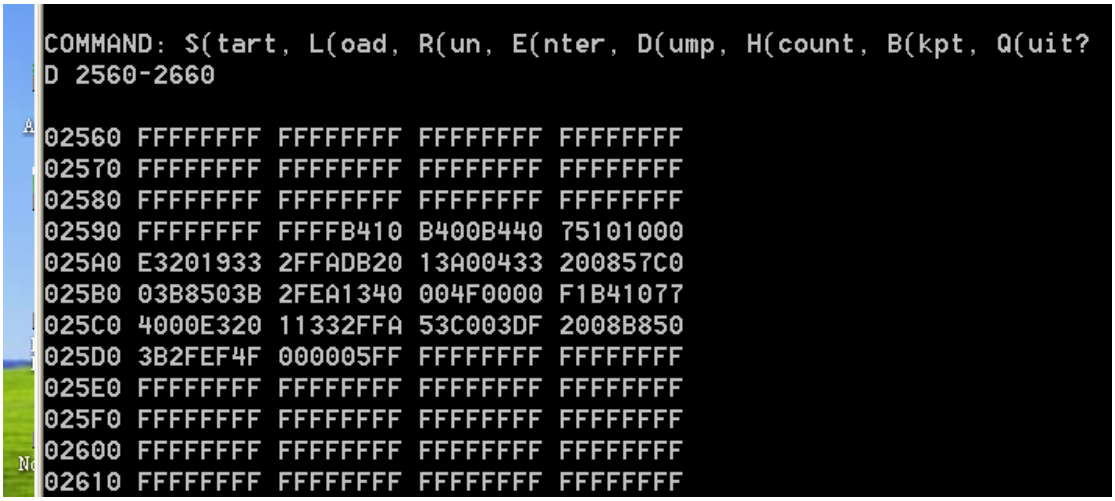
◆ Golden file

```
L
× Loading object code program ...  [HCOPY  000000001077      ]

  Total Object Code Records: 10

   + Type of object code: Relocatable object code (001560)
   + Program/Image NCOPY
   + Starting Memory Address: 000000
   + Size of OBJ Program/Image: 001077
   + Transfer Address: 000000  (001560)

× OBJ Code Program/Memory Image Loaded.
```

Start address(relocatable): 001560

因此可以 dump 001560-001660 和 002560-002660 的 image file 出來看

```
COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 1560-1660

01560 17202D69 202D4B10 25960320 26290000
01570 3320074B 1025BD3F 2FEC0320 100F2016
01580 0100030F 200D4B10 25BD3E20 03454F46
01590 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
015A0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
015B0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
015C0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
015D0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
015E0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
```


COMMAND: S(tart, L(oad, R(un, E(nter, D(ump, H(count, B(kpt, Q(uit?
D 2560-2660
02560 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02570 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02580 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02590 FFFFFFFF FFFFB410 B400B440 75101000
025A0 E3201933 2FFADB20 13A00433 200857C0
025B0 03B8503B 2FEA1340 004F0000 F1B41077
025C0 4000E320 11332FFA 53C003DF 2008B850
025D0 3B2FEF4F 000005FF FFFFFFFF FFFFFFFF
025E0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
025F0 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02600 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
02610 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF