

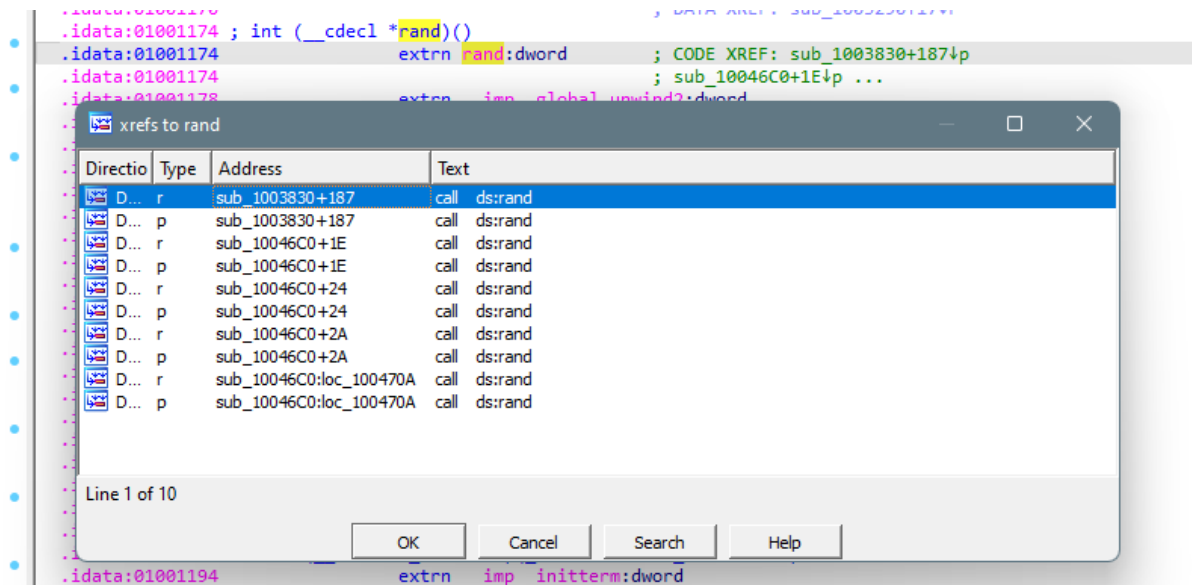
This document shares how our group found the cards in memory:

a. As each game randomly places all the cards, it makes sense that the cards would be generated near a rand function

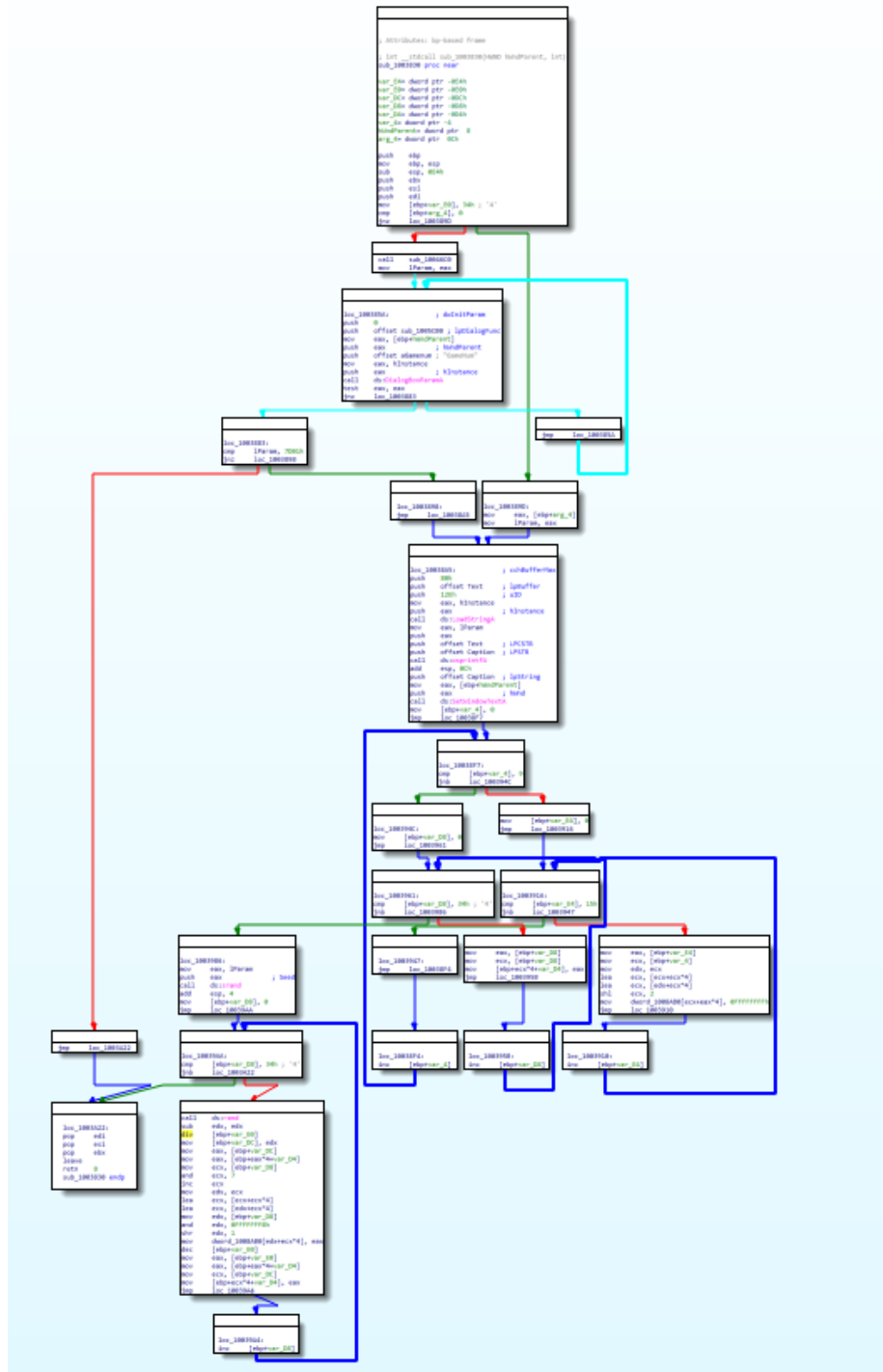


000000000100115C	EnableMenuItem	USER32
0000000001001160	InvalidateRect	USER32
0000000001001168	__p__pctype	msvcrt
000000000100116C	__isctype	msvcrt
0000000001001170	__p__mb_cur_max	msvcrt
0000000001001174	rand	msvcrt
0000000001001178	__global_unwind2	msvcrt
000000000100117C	srand	msvcrt
0000000001001180	time	msvcrt
0000000001001184	_exit	msvcrt
0000000001001188	XrptFilter	msvcrt

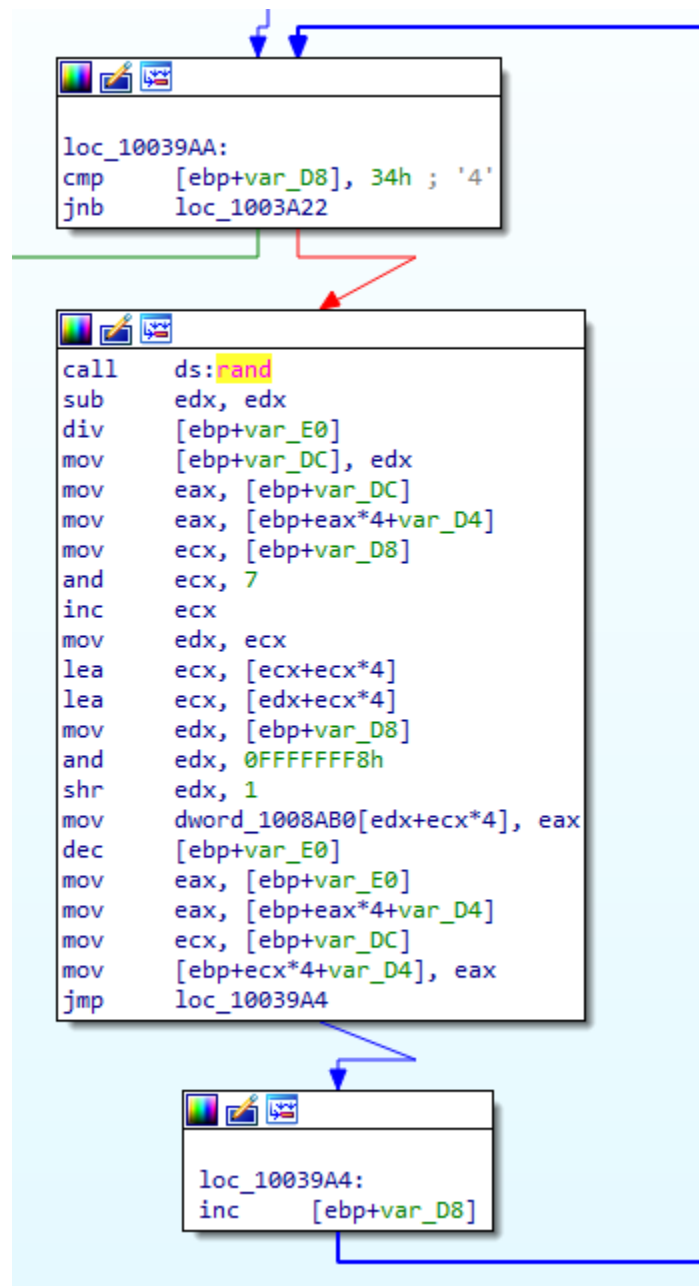
2. Find where rand is called using xrefs



3. Clicking on the first entry above we are brought to the following code



... which contains the following loop:



... notice this loop continues “34h” times which is 52 in decimal. Also, it calls the rand function at the top. This seems like a place where cards are being generated.

- Specifically, after analyzing the code it appears the card is being placed into dword_1008AB0. This instruction occurs at 010039F6:

```
mov     dword_1008AB0[edx+ecx*4], eax
```

```
100.00% (2,1688) (365,934) 000039F6 00000000010039F6: sub_1003830+1C6 (Synchronized with Hex View-1)
```

- Moving to OllyDbg we go to that instruction and set a breakpoint

OllyDbg - freecell.exe

File View Debug Plugins Options Window Help

CPU - main thread, module freecell

010038E8	. C745 FC 000000	MOV DWORD PTR SS:[EBP-4],0	
010038EF	.vE9 03000000	JMP freecell.010038F7	
010038F4	> FF45 FC	INC DWORD PTR SS:[EBP-4]	
010038F7	> 837D FC 09	CMP DWORD PTR SS:[EBP-4],9	
010038FB	.v0F83 4B000000	JNB freecell.0100394C	
01003901	. C785 1CFFFFFF	MOV DWORD PTR SS:[EBP-E4],0	
01003908	.vE9 06000000	JMP freecell.01003916	
01003910	> FF85 1CFFFFFF	INC DWORD PTR SS:[EBP-E4]	
01003916	> 83BD 1CFFFFFF	CMP DWORD PTR SS:[EBP-E4],15	
0100391D	.v0F83 24000000	JNB freecell.01003947	
01003923	. 8B85 1CFFFFFF	MOV EAX,DWORD PTR SS:[EBP-E4]	
01003929	. 8B4D FC	MOV ECX,DWORD PTR SS:[EBP-4]	
0100392C	. 8BD1	MOV EDI,ECX	
0100392E	. 8D0C89	LEA ECX,DWORD PTR DS:[ECX+ECX*4]	
01003931	. 8D0C8A	LEA ECX,DWORD PTR DS:[EDX+ECX*4]	
01003934	. C1E1 02	SHL ECX,2	
01003937	. C78481 B08A0001	MOV DWORD PTR DS:[ECX+EAX*4+1008AB0],EAX	
01003942	.vE9 C9FFFFFF	JMP freecell.01003910	
01003947	>vE9 A8FFFFFF	JMP freecell.010038F4	
0100394C	> C785 28FFFFFF	MOV DWORD PTR SS:[EBP-D8],0	
01003956	.vE9 06000000	JMP freecell.01003961	
0100395B	> FF85 28FFFFFF	INC DWORD PTR SS:[EBP-D8]	
01003961	> 83BD 28FFFFFF	CMP DWORD PTR SS:[EBP-D8],34	
01003968	.v0F83 18000000	JNB freecell.01003986	
0100396E	> 8B85 28FFFFFF	MOV EAX,DWORD PTR SS:[EBP-D8]	
01003974	> 8B8D 28FFFFFF	MOV ECX,DWORD PTR SS:[EBP-D8]	
0100397A	. 89848D 2CFFFFFF	MOV DWORD PTR SS:[EBP+ECX*4-D4],EAX	
01003981	.vE9 D5FFFFFF	JMP freecell.0100395B	
01003986	> A1 48810001	MOV EAX,DWORD PTR DS:[10081481]	
0100398B	. 50	PUSH EAX	
0100398C	. FF15 7C110001	CALL DWORD PTR DS:[<&msvort srand>]	seed => 0 rand
01003992	. 83C4 04	ADD ESP,4	
01003995	. C785 28FFFFFF	MOV DWORD PTR SS:[EBP-D8],0	
0100399F	.vE9 06000000	JMP freecell.010039A9	
010039A4	> FF85 28FFFFFF	INC DWORD PTR SS:[EBP-D8]	
010039A9	> 83BD 28FFFFFF	CMP DWORD PTR SS:[EBP-D8],34	
010039B1	.v0F83 6B000000	JNB freecell.01003A22	
010039B7	. FF15 74110001	CALL DWORD PTR DS:[<&msvort rand>]	rand
010039BD	. 2BD2	SUB EDI,EDI	
010039BF	. F7B5 20FFFFFF	DIV DWORD PTR SS:[EBP-E0],EDI	
010039C5	. 8995 24FFFFFF	MOV DWORD PTR SS:[EBP-DC],EDI	
010039CB	. 8B85 24FFFFFF	MOV EAX,DWORD PTR SS:[EBP-DC]	
010039D1	. 8B8485 2CFFFFFF	MOV EAX,DWORD PTR SS:[EBP+EAX*4-D4]	
010039D8	. 8B8D 28FFFFFF	MOV ECX,DWORD PTR SS:[EBP-D8]	
010039DE	. 83E1 07	AND ECX,7	
010039E1	. 41	INC ECX	
010039E2	. 8BD1	MOV EDI,ECX	
010039E4	. 8D0C89	LEA ECX,DWORD PTR DS:[ECX+ECX*4]	
010039E7	. 8D0C8A	LEA ECX,DWORD PTR DS:[EDX+ECX*4]	
010039EA	. 8B95 28FFFFFF	MOV EDI,DWORD PTR SS:[EBP-D8]	
010039F0	. 83E2 F8	AND EDI,FFFFFFF8	
010039F3	. C1E1 01	SHR EDI,1	
010039F6	. 89848A B08A0001	MOV DWORD PTR DS:[EDX+ECX*4+1008AB0],EDI	
010039FD	. FF8D 20FFFFFF	DEC DWORD PTR SS:[EBP-E0]	
01003A03	. 8B85 20FFFFFF	MOV EAX,DWORD PTR SS:[EBP-E0]	
01003A09	. 8B8485 2CFFFFFF	MOV EAX,DWORD PTR SS:[EBP+EAX*4-D4]	
01003A10	. 8B8D 24FFFFFF	MOV ECX,DWORD PTR SS:[EBP-DC]	
01003A16	. 89848D 2CFFFFFF	MOV DWORD PTR SS:[EBP+ECX*4-D4],EAX	
01003A1D	.vE9 82FFFFFF	JMP freecell.010039A4	
01003A22	> 5F	POP EDI	
01003A23	. 5E	POP ESI	
01003A24	. 5B	POP EBX	
01003A25	. C9	LEAVE	
01003A26	. C2 0800	RET 8	
01003A29	. CC	INT3	
01003A2A	. CC	INT3	
01003A2B	. CC	INT3	
01003A2C	. CC	INT3	
01003A2D	. CC	INT3	
01003A2E	. CC	INT3	
01003A2F	. CC	INT3	
01003A30	. 55	PUSH EBP	
01003A31	. 8BEC	MOV EBP,ESP	
01003A33	. 83EC 5C	SUB ESP,5C	
01003A36	. 53	PUSH EBX	
01003A37	. 56	PUSH ESI	
01003A38	. 57	PUSH EDI	

... as we expect it occurs near the call to rand.

6. When we run Freecell with this breakpoint you will notice that a large area of memory is free:

[illegible]

... as we hit run to execute one loop, the memory begins to fill in:

After 1 loop:

[illegible]

After 15 more loops:

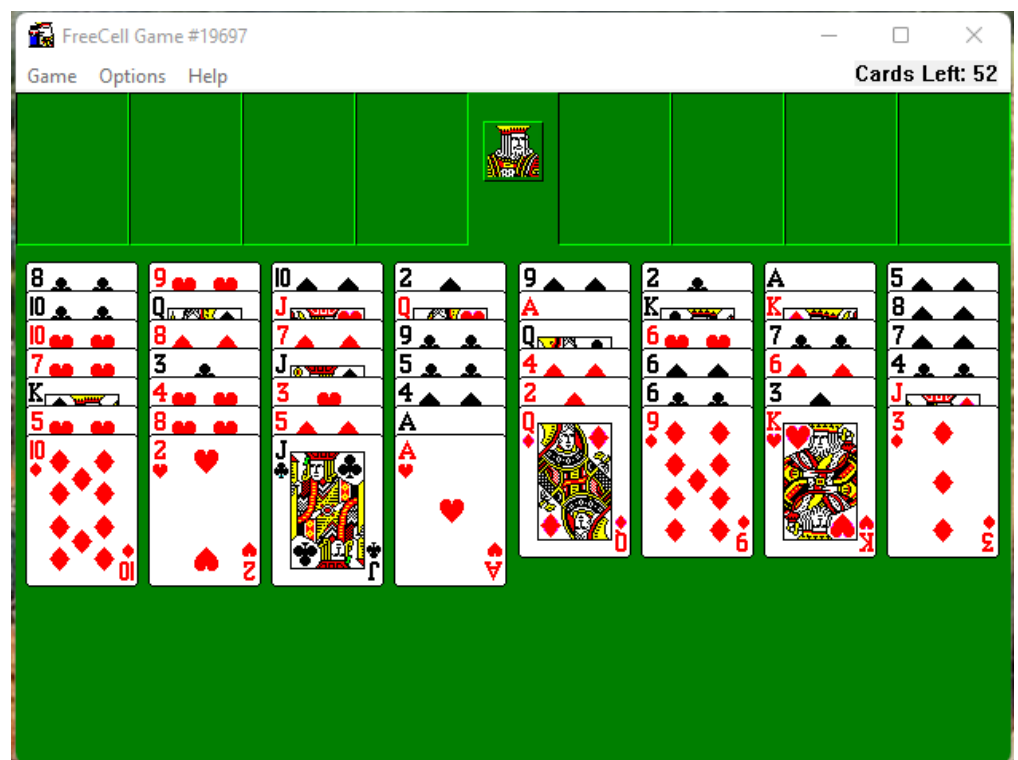
Address	Hex dump	ASCII
01008A98	00 00 00 00 00 00 00 00
01008A99	00 00 00 00 00 00 00 00
01008A9E	00 00 00 00 00 00 00 00
01008AA0	00 00 00 00 E1 13 05 416!4A
01008AA3	00 00 00 00 00 00 00 00
01008AB0	FF FF FF FF FF FF FF FF
01008AB5	FF FF FF FF FF FF FF FF
01008AC0	FF FF FF FF FF FF FF FF
01008AC8	FF FF FF FF FF FF FF FF
01008AD0	FF FF FF FF FF FF FF FF
01008AD8	FF FF FF FF FF FF FF FF
01008AE0	FF FF FF FF FF FF FF FF
01008AE8	FF FF FF FF FF FF FF FF
01008AF0	FF FF FF FF FF FF FF FF
01008AF3	FF FF FF FF FF FF FF FF
01008B00	FF FF FF FF 1C 00 00 00L...
01008B08	24 00 00 00 FF FF FF FF	\$....
01008B10	FF FF FF FF FF FF FF FF
01008B18	FF FF FF FF FF FF FF FF
01008B20	FF FF FF FF FF FF FF FF
01008B28	FF FF FF FF FF FF FF FF
01008B30	FF FF FF FF FF FF FF FF
01008B38	FF FF FF FF FF FF FF FF
01008B40	FF FF FF FF FF FF FF FF
01008B43	FF FF FF FF FF FF FF FF
01008B50	FF FF FF FF FF FF FF FF
01008B58	22 00 00 00 2F 00 00 00	".../...
01008B68	FF FF FF FF FF FF FF FF
01008B70	FF FF FF FF FF FF FF FF
01008B78	FF FF FF FF FF FF FF FF
01008B80	FF FF FF FF FF FF FF FF
01008B88	FF FF FF FF FF FF FF FF
01008B90	FF FF FF FF FF FF FF FF
01008B98	FF FF FF FF FF FF FF FF
01008BA0	FF FF FF FF FF FF FF FF
01008BA8	FF FF FF FF 27 00 00 00?
01008BB0	2A 00 00 00 FF FF FF FF	*...
01008BB8	FF FF FF FF FF FF FF FF
01008BC0	FF FF FF FF FF FF FF FF
01008BC8	FF FF FF FF FF FF FF FF
01008BD0	FF FF FF FF FF FF FF FF
01008BD8	FF FF FF FF FF FF FF FF
01008BE0	FF FF FF FF FF FF FF FF
01008BE8	FF FF FF FF FF FF FF FF
01008BF0	FF FF FF FF FF FF FF FF
01008BF8	FF FF FF FF FF FF FF FF
01008C00	07 00 00 00 2E 00 00 00	*.....
01008C08	FF FF FF FF FF FF FF FF
01008C10	FF FF FF FF FF FF FF FF
01008C18	FF FF FF FF FF FF FF FF
01008C20	FF FF FF FF FF FF FF FF
01008C28	FF FF FF FF FF FF FF FF
01008C30	FF FF FF FF FF FF FF FF
01008C38	FF FF FF FF FF FF FF FF
01008C40	FF FF FF FF FF FF FF FF
01008C48	FF FF FF FF FF FF FF FF
01008C50	FF FF FF 23 00 00 00#...
01008C58	01 00 00 00 FF FF FF FF	0... ..
01008C60	FF FF FF FF FF FF FF FF
01008C68	FF FF FF FF FF FF FF FF
01008C70	FF FF FF FF FF FF FF FF
01008C78	FF FF FF FF FF FF FF FF
01008C80	FF FF FF FF FF FF FF FF
01008C88	FF FF FF FF FF FF FF FF
01008C90	FF FF FF FF FF FF FF FF
01008C98	FF FF FF FF FF FF FF FF
01008CA0	FF FF FF FF FF FF FF FF
01008CA8	04 00 00 00 30 00 00 000...
01008CB0	FF FF FF FF FF FF FF FF
01008CB8	FF FF FF FF FF FF FF FF
01008CC0	FF FF FF FF FF FF FF FF
01008CC8	FF FF FF FF FF FF FF FF
01008CD0	FF FF FF FF FF FF FF FF
01008CD8	FF FF FF FF FF FF FF FF
01008CE0	FF FF FF FF FF FF FF FF
01008CE8	FF FF FF FF FF FF FF FF
01008CF0	FF FF FF FF FF FF FF FF
01008CF8	FF FF FF 03 00 00 00#...
01008D00	31 00 00 00 FF FF FF FF	1...
01008D08	FF FF FF FF FF FF FF FF
01008D18	FF FF FF FF FF FF FF FF
01008D20	FF FF FF FF FF FF FF FF
01008D28	FF FF FF FF FF FF FF FF
01008D30	FF FF FF FF FF FF FF FF
01008D38	FF FF FF FF FF FF FF FF
01008D40	FF FF FF FF FF FF FF FF
01008D48	FF FF FF FF FF FF FF FF
01008D58	13 00 00 00 1F 00 00 00	!...V...
01008D68	FF FF FF FF FF FF FF FF
01008D78	FF FF FF FF FF FF FF FF
01008D88	FF FF FF FF FF FF FF FF
01008D98	FF FF FF FF FF FF FF FF
01008DA8	FF FF FF FF FF FF FF FF
01008DB0	FF FF FF FF FF FF FF FF
01008DB8	00 00 00 00 00 00 00 00
01008DC0	00 00 00 00 00 00 00 00

... finally after 52 loops the memory is filled with 52 hex values that represent the cards

Address	Hex	dump	ASCII
01008A88	00	00 00 00 00
01008A90	00	00 00 00 00
01008A98	00	00 00 00 00
01008AA0	00	00 00 00 00E 1 13 05 41
01008AA8	00	00 00 00 00
01008AB0	FF	FF FF FF FF
01008AB8	FF	FF FF FF FF
01008AC0	FF	FF FF FF FF
01008AC8	FF	FF FF FF FF
01008AD0	FF	FF FF FF FF
01008AD8	FF	FF FF FF FF
01008AE0	FF	FF FF FF FF
01008AE8	FF	FF FF FF FF
01008AF0	FF	FF FF FF FF
01008AF8	FF	FF FF FF FF
01008B00	FF	FF FF FF 1CL...
01008B08	24	00 00 00 26\$...
01008B10	1A	00 00 00 33+...3...
01008B18	12	00 00 00 25@...%
01008B20	FF	FF FF FF FF
01008B28	FF	FF FF FF FF
01008B30	FF	FF FF FF FF
01008B38	FF	FF FF FF FF
01008B40	FF	FF FF FF FF
01008B48	FF	FF FF FF FF
01008B50	FF	FF FF FF FF
01008B58	22	00 00 00 2F"....
01008B60	1D	00 00 00 08#...D...
01008B68	0E	00 00 00 1EB...A...
01008B70	06	00 00 00 0F@...
01008B78	FF	FF FF FF FF
01008B80	FF	FF FF FF FF
01008B88	FF	FF FF FF FF
01008B90	FF	FF FF FF FF
01008B98	FF	FF FF FF FF
01008BA0	FF	FF FF FF FF
01008BA8	FF	FF FF FF 27'...
01008BB0	2A	00 00 00 19*...+...
01008BB8	2B	00 00 00 0A+.....
01008BC0	11	00 00 00 28<...(...
01008BC8	FF	FF FF FF FF
01008BD0	FF	FF FF FF FF
01008BD8	FF	FF FF FF FF
01008BE0	FF	FF FF FF FF
01008BE8	FF	FF FF FF FF
01008BF0	FF	FF FF FF FF
01008BF8	FF	FF FF FF FF
01008C00	07	00 00 00 2E"....
01008C08	20	00 00 00 10@...D...
01008C10	0F	00 00 00 00*.....
01008C18	FF	FF FF FF FF
01008C20	FF	FF FF FF FF
01008C28	FF	FF FF FF FF
01008C30	FF	FF FF FF FF
01008C38	FF	FF FF FF FF
01008C40	FF	FF FF FF FF
01008C48	FF	FF FF FF FF
01008C50	FF	FF FF FF 23#...
01008C58	01	00 00 00 2C@.....
01008C60	0D	00 00 00 05@...@...
01008C68	2D	00 00 00 FF-...
01008C70	FF	FF FF FF FF
01008C78	FF	FF FF FF FF
01008C80	FF	FF FF FF FF
01008C88	FF	FF FF FF FF
01008C90	FF	FF FF FF FF
01008C98	FF	FF FF FF FF
01008CA0	FF	FF FF FF FF
01008CA8	04	00 00 00 30@...@...
01008CB0	16	00 00 00 17@...@...
01008CB8	14	00 00 00 21@...@...
01008CC0	FF	FF FF FF FF
01008CC8	FF	FF FF FF FF
01008CD0	FF	FF FF FF FF
01008CD8	FF	FF FF FF FF
01008CE0	FF	FF FF FF FF
01008CE8	FF	FF FF FF FF
01008CF0	FF	FF FF FF FF
01008CF8	FF	FF FF FF 03@...
01008D00	31	00 00 00 181...+...
01008D08	15	00 00 00 08S...@...
01008D10	32	00 00 00 FF2...
01008D18	FF	FF FF FF FF
01008D20	FF	FF FF FF FF
01008D28	FF	FF FF FF FF
01008D30	FF	FF FF FF FF
01008D38	FF	FF FF FF FF
01008D40	FF	FF FF FF FF
01008D48	FF	FF FF FF FF
01008D50	13	00 00 00 1F!!...V...
01008D58	1B	00 00 00 0C+.....
01008D60	29	00 00 00 09).....
01008D68	FF	FF FF FF FF
01008D70	FF	FF FF FF FF
01008D78	FF	FF FF FF FF
01008D80	FF	FF FF FF FF
01008D88	FF	FF FF FF FF
01008D90	FF	FF FF FF FF
01008D98	FF	FF FF FF FF
01008DA0	FF	FF FF FF 00
01008DA8	00	00 00 00 00
01008DB0	00	00 00 00 00
01008DB8	00	00 00 00 00

7. With this we can then make a mapping from the hex values to there corresponding card by using the cards displayed on the Freecell GUI and the cards in memory represented by hex:

```
FF FF FF FF FF FF FF FF
FF FF FF FF 1C 00 00 00
24 00 00 00 26 00 00 00
1A 00 00 00 33 00 00 00
12 00 00 00 25 00 00 00
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
22 00 00 00 2F 00 00 00
10 00 00 00 08 00 00 00
0E 00 00 00 1E 00 00 00
06 00 00 00 FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
27 00 00 00 19 00 00 00
2A 00 00 00 0A 00 00 00
28 00 00 00 28 00 00 00
11 00 00 00 FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
07 00 00 00 2E 00 00 00
20 00 00 00 10 00 00 00
0F 00 00 00 00 00 00 00
02 00 00 00 FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
04 00 00 00 30 00 00 00
16 00 00 00 17 00 00 00
14 00 00 00 21 00 00 00
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF
13 00 00 00 1F 00 00 00
1B 00 00 00 0C 00 00 00
29 00 00 00 09 00 00 00
FF FF FF FF FF FF FF FF
```



8. From this we were able to make the following mapping from hex to card. Check for yourself that the cards match. Notice that the 1st clump of cards corresponds to the 1st column of cards, the 2nd clump to the 2nd column, and so on. Also note that the last hex value in a “clump” corresponds to the top card of a column:

	A	B	C	D	E
1	# \ Suit	Clubs	Diamonds	Hearts	Spades
2	A	00	01	02	03
3	2	04	05	06	07
4	3	08	09	0A	0B
5	4	0C	0D	0E	0F
6	5	10	11	12	13
7	6	14	15	16	17
8	7	18	19	1A	1B
9	8	1C	1D	1E	1F
10	9	20	21	22	23
11	10	24	25	26	27
12	J	28	29	2A	2B
13	Q	2C	2D	2E	2F
14	K	30	31	32	33

9. We have successfully located the cards and we know how they are represented! Now we just need a program that can read those addresses of memory.