ROM 化プログラム(RAM に配置するプログラム)の持ち方

ROM 化するプログラムはロード・モジュールと HEX オブジェクトで持ち方が異なっています。

HEX オブジェクトの場合には, RAM に配置するプログラムは_rcopy()関数の直後に出力されます。 下図は RAM に配置するプログラムの中身のアセンブリ言語でのリストと出力された HEX オブジェクトの対比です。左側のリストの最初で@@CODER となっていることから RAM 配置であることが分かります。

このプログラムのオブジェクトコードを見ると、"C7"、"20"、"04"、"FB"、"F8"、"FF"・・・となっています。HEX オブジェクトで探してみると 00532 番地から格納されています。

ここで,0049C 番地が_rcopy()関数で,その後ろに_rcopy()関数が対象にする RAM 配置するプロ グラムが入っていることがわかります。



RAM 配置のプログラムを MAP ファイルで確認すると, FE9C4 番地に配置されていることが分か ります。_rcopy()関数を実行することで, プログラムが実際の配置アドレス(FE9C4H)にコピーさ れます。____

@@CODER			FE9C4H	00022H	CSEG↓
	@@CODER	self tmp	FE9C4H	00022H↓	

では、通常デバッグで使用するロード・モジュールではどうなっているかと言うと、これとは少 し異なっています。

デバッガのプロパティで「ダウンロード・ファイル設定」タグの「ダウンロードするファイル」 で確認すると,次頁に示すようにロード・モジュールが一つ選択されて,そこで,「オブジェクト」, 「シンボル」をダウンロードような設定になっています。

 ダウンロード ダウンロードはるファイル ダウンロードはるファイル ダウンロードが後にCPUをリセットする ダウンロードがにフラッシュ ROMを消去す。 イベント設定位置の自動変更方法 予約領域の上書きをチェックする ・デバッグ情報 	ダウンロード・ファイル ダウンロード・ファイル一覧(<u>F</u>): G14UART.Imf	[1 /オ ダウンロード・ファイルのプロパティ(<u>P</u>): 4 ダウンロード・ファイル情報 ファイル DefaultBuild¥G14UART.It
CPUリセット後に指定シンボル位置まで 指定シンボル スタートアップ開始シンボル スタートアップ終了シンボル		ファイルの種類 ロード・モジュール・ファイル オブジェクトをダウン」はい シンボルをダウンロ はい 入力補完機能用のはい
	注意加(<u>A)</u> 肖明瑜(<u>R</u>)	ファイル ダウンロードするファイルを指定します。 OK キャンセル ヘルプ(<u>H</u>)
ダウンロードするファイル ダウンロードするファイルを指定します。[_]ボタ	ンを押下するとダウンロード・ファイル ダイアログオ	が開きます。 ダウンロード・ファイル ダイアログで、 ダウンロー

この状態でデバッガにダウンロードした結果を下に示します。049C 番地には何も入っていません。 また, RAM の FE9C4H 番地からは RAM に配置したプログラムが既に入っています。

Γ	/ 動逆アセン	/ブル1 🛛 r_ce_ser	ial.c!/ 🕋 プロパテ	ィ/ 🗹 self_tmp.c/ 📝 r_cg_port ₹ 🖣	×	x メモリ1 -	ļ.
	2 🚯 🜛	🦉 🕌 表示(I	.)+			2 ※ 表記(N) · サイズ表記(Z) · エンコード(C) ·	
	00497 00499	efe9 effe	BR BR	\$_exit+0x3 \$_exit+0x1a	-	▲ □ 停止時に移動 self_tmp 移動	
Ŀ	0049b	ff rcopy:	?			+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +	<u>- c</u>
E	0049c	ff	?			fe9c0 00 00 00 00 00 00 00 00 00 00 00 00 0	<u>3C</u>
Ŀ	0049e	ff	ż			fe9d0 4C 64 DE 0E 8B 0E 03 70 8C 01 1C 00 1	3B
Ŀ	00491 004a0	ff	2			fe9f0 00 00 00 00 00 00 00 00 00 00 00 00 0)0
	004a1	tt.	ž.			fea00 00 00 00 00 00 00 00 00 00 00 00 00)0(

このまま単純に実行すると,_rcopy()関数のところで停止してしまいます。この状態を避けるには, デバッグ時には_rcopy()関数の呼び出しをコメントアウトしておくか,ダウンロードの設定を変更 します。

具体的には、オブジェクトはヘキサ・オブジェクトをダウンロードし、ロード・モジュールから はシンボルだけをダウンロードします。

デバッガのプロパティで「ダウンロード・ファイル設定」タグの「ダウンロードするファイル」 画面を開きます。そこで、右の方の 📖 (下図参照)をクリックして、次頁に示す設定画面を開き ます。

⊿ ダウンロード		
▶ ダウンロードするファイル	[1]	
ダウンロード後にCPUをリセットする	(\$()	
Kova, P. T. Pater	コロ、い活牛	

ダウンロード・ファイル		—
ダウンロード・ファイル一覧(F): G14UART.Imf	(U) ↓(D)	ダウンロード・ファイルのプロパティ(P): ダウンロード・ファイル情報 ファイル DefaultBuild¥G14UART.In ファイルの種類 ロード・モジュール・ファイル オブジェクトをダウンIはい シンボルをダウンロ~はい 入力補完機能用のはい

ここで、下の図のように、オブジェクトをダウンロードの部分が「はい」になっているのを「い いえ」に変更します。

and the second and the second s		101.0
ダウンロード・ファイル		83
ダウンロード・ファイル一覧(F):		ダウンロード・ファイルのプロパティ(P):
G14UART.Imf	[↑(W)]	▲ ダウンロード・ファイル情報
		ファイル DefaultBuild¥G14UART.Ir
		ファイルの種類 ロード・モジュール・ファイル
	(オブジェクトをダウン(はい 📃
		シンボルをダウンロー(はい
		入力補完機能用ないいえ

次に左下の「追加(A)」をクリックしてダウンロード・ファイル一覧に「-」を追加します。「-」 が選択された状態で、右側の「ファイルの種類」で「ヘキサ・ファイル」を選択します。

<u></u>		
ダウンロード・ファイル		8
ダウンロード・ファイル一覧(F):		ダウンロード・ファイルのプロパティ(P):
G14UART.Imf	(U) ↑	▲ ダウンロード・ファイル情報
-		ファイルの種類 ード・モジュール・ファイル
	L + (D)	ロード・モジュール・ファイル
		へキリ・ファイル ヘキサ・ファイル(フラッシュプログラマ形式)
		バイナリ・データ・ファイル
		ファイルの理想 ダウンロードするファイルの種類を選択します。
追加(A) 削除(R)		_
		OK キャンセル ヘルプ(H)

次に,右の「ファイル」でダウンロードする HEX オブジェクトを選択します。下の例では, G14UART.hex を選択しています。

してロッピットT&LC TBJE 指定シンボル スタートアップ開始シ	シンパルロロはて天口 90 ダウンロード・ファイル	ldiu I	8	
スタートアップ終了シ	ダウンロード・ファイルー覧(F): G14UART.Imf - (U) (D) (D) (D) (D) (D) (D) (D) (D) (D) (D	ダウンロード・ファイルのプロパティ(P):	- 1 ,	
ダウンロードするファイル	を選択			×
CSC ♥ ↓ « CSS)	G14UART > DefaultBuild	✓ 4 Defaul	ltBuildの検索	Q
整理 ▼ 新しいフォノ	レダー			0
אעעב‡א 🔝	1 名前	更新日時	種類	サイス
■ ピクチャ	G14UART.hex	2014/04/01 9:23	HEX ファイル	1
Non Mark				

この状態で, E1 にダウンロードさせてみます。その結果が次ページの図になります。main 関数の実行前の段階では RAM の内容は不定になっています。

	* Arguments : None	f e9c0	00	00	00		F <u>2</u>	ĒF	08	00	E2	ĒF	04	91	DF
Lak	* Recurn value • Nune	feydu	UU	£5	A I	ין עַט	ςī.	<u>C3</u>	Ü5 -	UΤ	18E	ΕÇ.	10	8E	I F U
	* * * * * * * * * * * * * * * * * * * *	t e9eU	BD	28	41	0E (CE	FC	00	00	00	00	00	00	100
	void main(void)	fe9f0	00	00	00	00 0	00	00	00	00	00	00	00	00	00
		fea00	00	00	00	00 (00	00	00	00	00	00	00	00	00
	K_MAIN_USERINIt();	feal0	00	00	00	00 (00	00	00	00	00	00	00	00	00
1	/* Start user code. Do not edit comment generated here */	fea20	00	00	00	00 (00	00	00	00	00	00	00	00	00
	uinti6 t rot:	f ea30	00	00	00	00 (00	00	00	00	00	00	00	00	100
1	unitio_t ret,	fea40	00	00	00	00 (00	00	00	00	00	00	00	00	00
ុ៣,	ret = rcopy(1):	fea50	00	00	00	00 (00	00	00	00	00	00	00	00	00
		f ea60	00	00	00	00 L(00	00	00	00	100	00	00	00	100

この状態から, _rcopy()関数まで実行させてみます。下の図で RAM 領域の内容が RAM に配置するプログラムに書き換わっています。

00225	Vescription - This function imprements main function. * Arguments : None * Return Value : None *********************************	fe9ct 00
0022b	R_MAIN_UserInit(); /* Start user code. Do not edit comment generated here */ uint8_t work; uint16_t ret;	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$
00232 00235	<pre>@ ret = _rcopy(1);</pre>	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$

なぜか,235H 番地の_rcopy()関数の実行前でブレークしているように見えますが,逆アセンブル 画面で見ると,次の R_UART0_Start 関数の実行前でブレークしているのが分かります。つまり, _rcopy()関数はきちんと実行されて,プログラムは RAM に配置されているのが分かります。

0022c 00230 00231	fc9c0400 CALL 13 MOVW bb MOVW	 АХ,ВС ГН 1.АХ	fe9c0 fe9d0	UU UU <u>00</u> 00 <mark>40 64</mark>	UU U <u>OO</u> O DE O	JU UU U) 0 <u>C7</u> 2) E 8B 0	U UU O O4 E O3	UU UL FB F8 70 80	UU FF 01	<u>UU</u> F6 10	UU <u>UU</u> BB 9C 00 BB
73: 74:	uint16_t ret;	[IIC] (BA	fe9e0 fe9f0	EF EC	10 0 00 0		7 00			00	
00232	717bfa DI		fea00	00 00	ŏŏ ŏ	jo oo o	ŏ ŏŏ		i öö	ÖÖ	00 00
10: 00005	ret = _rcopy(I);		fea10	00 00	00 0	0 00 00	0 00	00 00	00 0	00	00 00
70.00233	TC270300 CALL	:: <u>K_UARIU_</u> Start	fea20	00 00	00 0	0 00 0	0 00	00 00	00	00	00 00

_rcopy()関数の部分を逆アセンブル画面で眺めると、ちゃんと_rcopy()関数が入っているのも確認できます。

	rcopy:			*	□ 停止問	上時に移動 self tmp 移動
UU49c	cb	PUSH	DE			
0049d	ç7	PUSH	HL			+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f
0049e	14	MOAM	DE,AX		00490	02 9E EC ABIC6 61 CA FEIE9 EE EE EE C5 C7 14 8E
00491	8etd	MUV	A,ES		004a0	ED C1 41 00 11 DB 28 05 E6 43 DD 74 B1 45 DD 04
004a1	CT /100	NOV	AA FS #04		00460	13 45 DC 6C 51 00 36 24 05 E0 80 C5 C3 C7 C1 C1
004a2	11db2805	MOVW	BC.ES: 1528H		004c0	0 F6 B1 45 C0 DD 06 C1 F1 45 C0 DF 34 9E FD 11 AB
004a8	f6	ÖLRŴ	AX		004d0	0 12 11 AC 02 14 11 AC 04 C1 11 AC 06 9E FD C6 C1
004a9	43	CMPW	AX,BC		004e0	0 11 88 08 9E FD 08 11 99 A7 F6 47 CO 61 F8 81 9E
004aa	dd74	BZ	\$ <u></u> rcopy+0x84		004f0	D FD C1 A5 F6 45 C0 61 F8 80 C1 B3 F6 43 C0 DF DF
UU4ac	bl	DECW	AX		0.0500	C4 C0 04 08 00 16 61 08 85 C2 83 F6 43 15 C4 00
UU4ad	45	CWLA	AX,UE		00510	1 1 C1 ER B1 45 C0 DD A2 C1 E1 45 C0 DE 90 EE 02
UU4ae	ddU4	BZ	\$ rcopy+Ux18		00010	P II CI FO BI 43 CO DD AZ CI FI 43 CO DF 3C EF 02