Travaux dirigés (Systèmes à processeurs n°1)

Analyse, codage et création de programmes simples

Exercice n°1 : Analyse d'un programme

On donne le programme source assembleur Renesas M32C87 suivant :

Etiquette	Opération	Opérande	Commentaire
VTMP	.EQU	1000h	//Valeur de la temporisation
	MOV.W:G	#VTMP,R1	
_JQCQ	ADD.W:G	# -1,R1	
	JNE	_JQCQ	

1. Donner une description algorithmique du programme
2. En utilisant les feuilles de codage fournies par le constructeur, coder le programme
3. Le microprocesseur fonctionne avec une fréquence d'horloge de 32MHz, évaluer le temps d'exécution du programme.

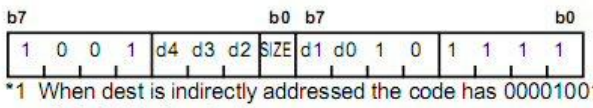
Exercice n°2 :

2.1. Transformer le programme de la temporisation en un sous programme dont le temps d'exécution est de 1 ms.

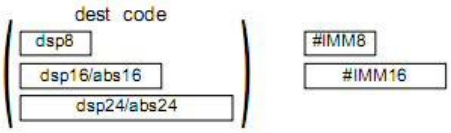
2.2. On utilise ce sous-programme pour construire un utilitaire réalisant une temporisation paramétrable. La durée de la temporisation sera de n ms. Le paramètre n sera chargé dans le registre R0.B

- Proposer un algorithme pour réaliser cet utilitaire.
- Coder l'algorithme en langage assembleur Renesas M32C87
- Quelle est la durée maximum de la temporisation ?

(1) MOV.size:G #IMM,dest



*1 When dest is indirectly addressed the code has 00001001 added at the beginning.



.size	SIZE
.B	0
.W	1

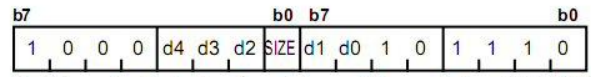
dest		d4 d3 d2 d1 d0	dest		d4 d3 d2 d1 d0
Rn	R0L/R0/---	1 0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	0 0 1 1 0
	R1L/R1/---	1 0 0 1 1		dsp:8[FB]	0 0 1 1 1
	R0H/R2/-	1 0 0 0 0	dsp:16[An]	dsp:16[A0]	0 1 0 0 0
	R1H/R3/-	1 0 0 0 1		dsp:16[A1]	0 1 0 0 1
An	A0	0 0 0 1 0	dsp:16[SB/FB]	dsp:16[SB]	0 1 0 1 0
	A1	0 0 0 1 1		dsp:16[FB]	0 1 0 1 1
[An]	[A0]	0 0 0 0 0	dsp:24[An]	dsp:24[A0]	0 1 1 0 0
	[A1]	0 0 0 0 1		dsp:24[A1]	0 1 1 0 1
dsp:8[An]	dsp:8[A0]	0 0 1 0 0	abs16	abs16	0 1 1 1 1
	dsp:8[A1]	0 0 1 0 1	abs24	abs24	0 1 1 1 0

[Number of Bytes/Number of Cycles]

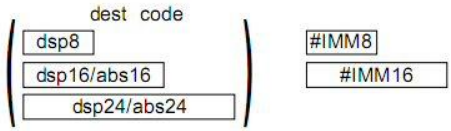
dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB/FB]	dsp:24[An]	abs16	abs24
Bytes/Cycles	3/1	3/1	3/2	4/2	4/2	5/2	5/2	6/2	5/2	6/2

*2 When dest is indirectly addressed, the number of bytes and cycles in the table are increased by 1 and 3 respectively.
*3 When (.W) is specified for the size specifier(.size) the number of bytes in the table is increased by 1.

(1) ADD.size:G #IMM,dest



*1 When dest is indirectly addressed the code has 00001001 added at the beginning.



.size	SIZE
.B	0
.W	1

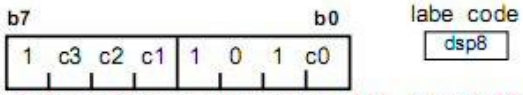
dest		d4 d3 d2 d1 d0	dest		d4 d3 d2 d1 d0
Rn	R0L/R0/---	1 0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	0 0 1 1 0
	R1L/R1/---	1 0 0 1 1		dsp:8[FB]	0 0 1 1 1
	R0H/R2/-	1 0 0 0 0	dsp:16[An]	dsp:16[A0]	0 1 0 0 0
	R1H/R3/-	1 0 0 0 1		dsp:16[A1]	0 1 0 0 1
An	A0	0 0 0 1 0	dsp:16[SB/FB]	dsp:16[SB]	0 1 0 1 0
	A1	0 0 0 1 1		dsp:16[FB]	0 1 0 1 1
[An]	[A0]	0 0 0 0 0	dsp:24[An]	dsp:24[A0]	0 1 1 0 0
	[A1]	0 0 0 0 1		dsp:24[A1]	0 1 1 0 1
dsp:8[An]	dsp:8[A0]	0 0 1 0 0	abs16	abs16	0 1 1 1 1
	dsp:8[A1]	0 0 1 0 1	abs24	abs24	0 1 1 1 0

[Number of Bytes/Number of Cycles]

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB/FB]	dsp:24[An]	abs16	abs24
Bytes/Cycles	3/1	3/1	3/3	4/3	4/3	5/3	5/3	6/3	5/3	6/3

*2 When dest is indirectly addressed, the number of bytes and cycles in the table are increased by 1 and 3 respectively.
*3 When (.W) is specified for the size specifier(.size) the number of bytes in the table is increased by 1.

(1) *Jcnd* label



dsp8 = address indicated by label - (start address of instruction +1)

Cnd	c3 c2 c1 c0	*Cnd*	c3 c2 c1 c0
LTU/NC	0 0 0 0	GEU/C	1 0 0 0
LEU	0 0 0 1	GTU	1 0 0 1
NE/NZ	0 0 1 0	EQ/Z	1 0 1 0
PZ	0 0 1 1	N	1 0 1 1
NO	0 1 0 0	O	1 1 0 0
GT	0 1 0 1	LE	1 1 0 1
GE	0 1 1 0	LT	1 1 1 0

[Number of Bytes/Number of Cycles]

Bytes/Cycles	2/1

*1 When branched to label the number of cycles in the table is increased by 2.