

LABORATÓRIOS MICROPROCESSADORES

KIT 8051 – Depuração e simulação.

Outubro 2006

1. Objectivo

Apresentar um guia básico de como usar o Keil µVision3 para simular e depurar programas em *assembly*.

2. Descrição

O ambiente µVision3 permite três formas diferentes de simulação:

a) Simulação passo-a-passo,

b) Simulação contínua com breakpoints,

c) Simulação contínua com actualização periódica do estado seleccionado do microcontrolador.

2.1 Criar um projecto

Crie e "assemble" o projecto abaixo, certificando que a configuração do debug é a seguinte:

Options for Target 'Target 1'	×			
Device Target Output Listing C51 A51 BL51 Lo © Use Simulator Settings Limit Speed to Real-Time	ocate BL51 Misc Debug Utilities			
Load Application at Startup Initialization File: Edit	Load Application at Startup Initialization File: Edit			
Restore Debug Session Settings Breakpoints IV Toolbox Watchpoints & PA Memory Display	Restore Debug Session Settings			
CPU DLL: Parameter: S8051.DLL	Driver DLL: Parameter:			
Dialog DLL: Parameter: DP51.DLL [-p51	Dialog DLL: Parameter: TP51.DLL -p51			
ОК Са	ncel Defaults Help			



🔽 teste - μVision3 - [C:\home\mic-8051\proj-	-1\teste.a51]				- 7 🛛				
Eile Edit View Project Debug Flash Peripherals	Tools SVCS Window Help				_ & ×				
爸 🖬 🕼 👃 요 🕸 🎼 🕰 요 % % 兆 嘛 📃 🖃 📕 # ← → 🐚 🔗 Q 📧 🗵 色 ‰ 15 200									
🕸 🎬 🎬 👗 🔤 🛣 Target 1									
Project Workspace • x 01 ; examp	ole for a p39-like pr	oblem							
Source Group 1 O3 Surce Group 1 O3 Surce Group 1 O4	cseg at O ljmp start								
05 06 table:	db 0,1,4,9,16,25,3	6,49							
07 08 09 start:	cseg at 100h mov p1,#0e0h	; make bits 5 t0 7 inputs							
10 11 loop:	mov a,p1	; read P1							
12 13	mov R2,#5	;shift right 5x (acc >> 5)							
15	djnz r2,rotlp								
17	anl a,#7 mov dptr,#table	;mask input bits ;get table address							
19 20 21	movc a,⊌a+dptr orl a #0e0b	get ith entry in table							
22 23	mov p1,a sjmp loop	<pre>;output to p1 ; repeat forever</pre>							
24 25	end								
26									
					•				
□ = \(\u0 *8 \\\\\\\0 \text{este.a51}									
* Build target 'Target 1'									
linking § Program Size: data=8.0 xdata=0	linking § Program Size: data=8.0 xdata=0 code=33								
Teste" - O Error(s), O Warnin	ng(s).				_				
Build & Command & Find in Files /	, 	Simulati	on	L:21 C:1	NUM R/W				

2.2 Active a janela de depuração "clicando" no *icon debug* e aparecerá a seguinte janela com a visualização dos registos:

🙀 teste - "pVision3 - [C:\home\mic-8051\proj=1\teste.n51]	I I I X
E Edit View Broket Beloup Flash Peophenelis Iools 2/CS Window Help	_ 8 ×
1911年月月1日日(休休人法法法理	
25 日本 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1	
Project Workspace • X 01 / example for a p39-like problem	-
Register Volue 02 cseet at 0	
10 0.00 ₽04 ljmp start	
-1 0.00 05 table: db 0,1,4,9,16,25,36,49	
- 13 0x00 07	
6 0x00 09 start: mov p1,#0=0h / make bits 5 t0 7 inputs	
- 15 0.00 10 - 7 0.00 11 loop: mov 0,pl / read Pl	
B-Sys 12 mm P2 #5 (200 5.5)	
b 600 14 rotlp: r a	
- sp 0.07 15 djnz r2,rotlp	
det 0.0000 17 and 9.97 mask ingut Site	
- HC \$ UNAUL. 10 move upter, resolve ypt i date date date date date date date date	
The count of the c	
22 mov pl,a joutput to pl	
24 simp 100p / repet zorever	
25 end 26	
	+
	<u>)</u>
Running with Code Size Limit: 2K "Load "C:NhomeNmic=805Thyproj=1hteste"	-
SASM ASSIGN BreakDisable BreakEnable BreakEnable BreakEith BreakList BreakSet BreakAccess COVERAGE DEFINE DIR Display Enter EVALuate	EXIT FUNC Go
S [4] S S S S S S S S S S	• • • • • • • • • • • • • • • • • • •
Newsy Simulation (1:0.00000000 see 1.221G)	R/W



A seta amarela indica a próxima instrução a ser executada ('ljmp start').

2.3 Abra a janela do porto 1 (P1), seleccionando o item do menu 'peripherals/IO Port/Port 1'

🕎 teste - µVisio	📅 teste - μVision3 - [C:\home\mic-8051\proj-1\teste.a51]										
Eile Edit View	Project <u>D</u> ebu	ug Fl <u>a</u> sh	Peripherals	<u>T</u> ools <u>S</u> V	CS <u>W</u> indow <u>H</u> elp						
1	X 🖻 🛍	<u>2</u> 2	Reset	⊆PU	5 % 🙀	★ # + → 12 ♣ Q					
Rst 🗐 🔕 🖓	∂+ {} + {}	\$₩	Intern	upt	. 🖬 🖿 🚟 🗖	×					
Project Workspace	* X	01	I/O- <u>P</u> o	orts 🔹 🕨	Port 0	oblem					
Register	Value	02	<u>S</u> erial		Port <u>1</u>						
- Regs		03	Timer	•	Port 2						
01	0x00	=>04		Հյութ	Port 3						
1	0x00	05		L	- Cont g						
r2	0x00	06	table:	db 0,	1,4,9,16,25,36	5,49					
r3	0x00	07									
r4	0x00	80		cseg	at 100h						
15	0x00	10	start:	mov	pi,#UeUn	; make bits 5 t0 7 inputs					
r6	0x00	11	loon	mou	a n1	, word Bi					
· · · · · · · · · · · · · · · · · · ·	UxUU	12	100p.	MOV	a,pi	/ reau Pi					
⊟ Sys	0.00	13		mov	R2 . #5	eshift right 5x (acc >> 5)					
a .	0x00	14	rotin:	rr	a	,Shirto right on (doc >> o)					
	0x00	15		dinz	r2.rotlp						
ep may	0x07	16		-							
dotr	0v0000	17		anl	a,#7	mask input bits					
PC \$	C:0x00	18		mov	dptr,#table	;get table address					
states	0	19		movc	a,0a+dptr	get ith entry in table					
sec	0.0000	20									
±	0x00	21		orl	\mathbf{a} , #OeOh	;make high order bits = 1					
		22		mov	p1,a	;output to p1					
		23		sjmp	loop	; repeat forever					
		24									
		25		end							
		26									

e visualizará a seguinte janela:

TT									
rester - pursion 3 - [C:\nomeunic=8051\proj-1\teste.a51]									
Ele Edit Wew Project Debug Flash Peripherals Iools SVCS Window Help									
12 😂 🖬 💋	አ 🖻 🛍	0		1 % %	× 🖏	→ # # = →	12 4	Q 🖪 🖪 🕂 🛞 🕅 🖉	
👫 🗒 🕅	0 [↓] () [↓] 1 ()	\$₩	02 🕅	💭 🖤 😼	🗆 🗄 🚟 🖪	بر			
Project Workspace	▲ X	01	; examp	le for a	p39-like pr	oblem			
Register	Value	02						Parallel Port 1	
🖃 Regs		03		cseg	at O			Port 1	
01	0x00	<204		ljmp	start			7 Bits 0	
f1	0x00	05	1000						
r2	0x00	05	tapie:	an 0,1,	,4,9,16,25,31	6,49		Pins: 0xFF	
- 13	0x00	07			at 100b				
r4	0x00	00	start:	mov	n1.#0e0h	· make hits 5 to 7 input	te		
ci ci	0.00	10		1101	<i>p1</i> //////	, make bits o to , input			
10	0x00	11	loop:	mov	a, p1	; read P1			
E Svs	01100	12							
a	0x00	13		mov	R2,#5	<pre>;shift right 5x (acc >></pre>	5)		
b	0x00	14	rotlp:	rr	a				
sp	0x07	15		djnz	r2, rotlp				
sp_max	0x07	16							
dptr	0x0000	1/		anı	a,#/	mask input bits			
PC \$	C:0x00	10		move	a Ra+dntr	get table address			
states	0 0000	20		1010	u) Sullaper	,get ith entry in table			
t now	0.0000	21		orl	a , #0e0h	:make high order bits =	1		
- porr	0,00	22		mov	p1,a	coutput to pl			
		23		sjmp	loop	; repeat forever			
		24							
		25		end					
		26							

2.4 Pressione F11 (depuração passo-a-passo) duas vezes e verifique:

- a) A localização da seta amarela moveu para a etiqueta '*loop*' e as duas instruções que foram executadas ficaram com uma marca a verde.
- b) A janela do porto 1 apresenta os três bits mais significativos a '1' e os restantes a '0'.



VZ t	🔽 teste - µVision3 - [C:\home\mic-8051\proj-1\teste.a51]										
	Elle Edit View Project Debug Flash Perjpherals Iools SVCS Window Help										
1											
				-	10 10 10	200 - 1941 J					
RST	器 国 ② ④ ① ① ① ③ 該 ③ ② ③ ② 参 話 □ 巨 圏 □ □ レ										
Proje	ct Workspace	* X	01	; exami	le for a	p39-like pr	oblem				
Rec	jister	Value	02	· · · ·				Parallel Port 1			
	Reas		03		cseg	at O		- Part 1			
	01	0x00	04		1jmp	start		7 Bits 0			
	r1	0x00	05								
	r2	0x00	06	table:	db 0,1	,4,9,16,25,3	6,49				
	r3	0x00	07								
	r4	0x00	80		cseg	at 100h					
	15	0x00	10	start:	mov	p1,#UeUn	; make bits 5 t0 7 inputs				
	r6	UXUU	P 11	loont	mour	a n1	· read B1				
	······································	UXUU	12	roop.	1110 1	a,pi	, ieau ri				
	Sys	0.00	13		mov	R2,#5	shift right 5x (acc >> 5)				
	a b	0x00	14	rotlp:	rr	a	,				
	sn.	0x07	15		djnz	r2, rotlp					
	sp max	0x07	16								
	dptr	0x0000	17		anl	a,#7	;mask input bits				
	PC \$	C:0x01	18		mov	dptr, #table	get table address;				
	states	4	19		MOAG	a,0a+dptr	get ith entry in table;				
	sec	0.0000	20		_						
	+ psw	0x00	21		orl	a, #UeUh	;make high order bits = 1				
			22		mov	p1,a	youtput to pl				
			23		ջյաք	TOOD	; repeat forever				
			24		end						
			25								

Nota:

Consideremos o pino 0 que está configurado como pino de saída, logo se tentar alterar o estado do pino externamente ('clicando' sobre o pino P1.0) receberá uma mensagem de erro. Numa implementação física, este procedimento resultaria na danificação do Porto 1.



2.5 Pressione o *icon* de *reset*, **S**, para repor o estado inicial do processador.



Testa	a3 - IC-\hom	almic.8	051\nroi-	1\tecte_a5	11				
Elle Edit Yew Project Debug Flash Perjaherals Iools SVCS Window Help									
12 🚔 🖬 🕼	X 🖻 🛍	00		16 % %	76 🖗	▼ 約 1件	← → ()2	a 🔍 🔼 🎮 🕁 🏀 🕅 🕅	
RST 🗉 🛛 (†)	0↓ {} * ()	\$₩	02 🔍	💭 🍼 😼	🗆 🗄 🚟 🖪	7			
Project Workspace	* X	01	; examp	le for a	p39-like pr	oblem			
Register	Value	02						Parallel Port 1	
E-Beas		03		cseg	at 0			Part 1	
10	0x00	➡ 04		ljmp	start			7 Bits 0	
- 1	0x00	05						P1: 0xFF	
	0x00	06	table:	db 0,1	,4,9,16,25,3	6,49		Dia D.C.	
13	0x00	07							
	0x00	08		oseg	at 100h				
	0x00	09	start:	mov	p1,#0e0h	; make bits 5 t0 7	inputs		
r6	0x00	10							
17	0x00	11	loop:	mov	a,pi	; read P1			
🖻 — Sys		12							
a	0x00	13		mov	R2,#5	;shift right 5x (ad	cc >> 5)		
b	0x00	14	rotlp:	rr	a				
sp	0x07	15		djnz	r2,rotlp				
sp_max	0x07	16							
dptr	0x0000	17		anl	a,#7	mask input bits			
PC \$	C:0x00	18		mov	dptr,#table	get table address;			
states	0	19		movc	a,∦a+dptr	get ith entry in t	table		
sec	0.0000	20							
	0x00	21		orl	a , #0e0h	;make high order bi	its = 1		
		22		mov	p1,a	joutput to pl			
		23		sjmp	100b	; repeat forever			
		24		22					
		25		end					
		26							
		11							

2.6 Seleccione a janela 'view/disassembly window'

V	teste	- μV	isio	n3 - [C:\home\mic-8051\proj-	1\teste.a5	1]		
	Ele	Edit [⊻jew	Project Debug Flash Peripherals	Iools SVCS	i <u>W</u> indow <u>H</u> elp		
1	1		✓ ✓	Status Bar File Toolbar	\$ % %	% 🖗 📃		Q 🖪 🚬 🕁 🏀 🕅 🖽
	RST 🗎	1 🛛		Build Toolbar	🔊 🌣 🛃	😐 🗄 🚟 🔽 ,	<u> </u>	
P	oject W	orkspa	~	Debug Toolbar	le for a	n39-like nro		
	Register Reg	s	•	Project Window	cseg	at 0		Parallel Port 1
L		rO	Þ	Output Window	1,3mp	start		P1: DVEE 7 Bits 0
l		r1 r2	(19	Sourc <u>e</u> Browser	db 0,1	4,9,16,25,36	5,49	
L		r4	<u>0</u>	Disassembly Window	cseg	at 100h		
L		r5		Watch & Call Stark Window	mov	p1,#0e0h	; make bits 5 t0 7 inputs	
L		r6 r7	<u> </u>	Memory Window	mov	a , p1	; read P1	
B	Sys	a	000E	Code Coverage Window Performance Analyzer Window	mov	R2, #5	<pre>;shift right 5x (acc >> 5)</pre>	
L		b sn		Logic Analyzer Window	rr djnz	a r2, rotlp		
L		sp m	5	Symbol Window				
L		dptr	5	Serial Window # <u>1</u>	anl	a,#7	mask input bits	
L		PC \$	3	Serial Window #2	mov	aptr,#table	get table address	
L		states	¥	Serial Window #3	110000	a, gatuper	yget ith entry in table	
L		DSW	ブ	Toolpox	orl	a , #0e0h	make high order bits = 1	
L			_		mov	p1,a	youtput to pl	
L			~	Periodic Window Update	sjmp	loop	; repeat forever	
			~	Include File Dependencies	end			

que visualizará o código assembly juntamente com o código máquina.

-							
¥4	teste -µVisior	n3 - [Disasse	embly]				
	<u>Eile E</u> dit <u>V</u> iew	Project Deb	ug Fl <u>a</u> sh Periph	ierals <u>T</u> ools <u>S</u> V	CS <u>W</u> indow <u>H</u> elp		
ł	🖹 🚅 🖩 🕼	X 🖻 🖻		傳 16 % %	5 % 🙀	▼ 桷 仲 →	(1) 🗇 🔍 🗖 🏹 🕁 🗞 🕅 🕅
G	87 🛛 🗉 🖓	0 • ()+ →()	◆ 凝출 ()숲	💽 🐺 🖤 🗄	s 🗆 🖹 🚟 🗖 .	<u>ج</u>	
Pre	oject Workspace	* X	4		l im:	o start	
B	Register	Value	5		- 51		Darallol Dort 1
Ē	Beas		6	: table:	db 0.	1,4,9,16,25,36,49	
		0×00	7	:			Port 7 Bite 0
		0×00	8	:	cses	7 at 100h	P1: 0xFF UUUUUUUU
		0x00	C:0x00	0201	LOO LJMP	START(C:0100)	Day OFF
	- 13	0x00	C:0x000	03 00	NOP		
	14	0x00	C:0x000	0104	1 AJMP	C:0004	
	r5	0x00	C:0x000	06 09	INC	R1	
	r6	0x00	C:0x000	07 1019	924 JBC	0x23.1,C:002E	
	17	0x00	C:0x000	DA 3100) ACALL	START(C:0100)	
È	- Sys		C:0x000	DC 00	NOP		
	a	0x00	C:0x000	DD 00	NOP		
	Ь	0x00	C:UxUU	JE UU	NOP		
	sp	0x07	C:0x000	JF UU	NOP		
	sp_max	0x07	C:0x00.	10 00	NOP		
	dptr	0x0000	C:UXUU.	11 00	NOP		
	PC \$	C:0x00	C:0x00.	12 00	NOP		
	states	0	C:0x00.	13 00	NOP		
	sec	0.0000	C:0x00	15 00	NOP		
		0x00	C:0x00.	15 00	NOP		
			C:0x00	17 00	NOP		
			C.0x00.	10 00	NOD		



Repare que a localização C:0000H contém o valor 02H, o C:0001H contém 01H, o C:0002H contém 00H ... Repare de igual modo que os *bytes* da tabela, 'table', foram cegamente "disassembladas" como se fossem instruções.

- 2.7 Ajuste a janela de registos de modo a que o acumulador seja visível, caso este esteja escondido.
- 2.8 Pressione F11 até que a instrução 'mov' na localização C:0103H seja executada.



Repare que o registo acumulador teve o seu conteúdo alterado para E0H e ficou colorido a azul (os registos cujos conteúdos foram alterados ao longo da execução são coloridos a azul).

Poderia continuar a pressionar F11 e executar uma instrução de cada vez, mas isto seria contraproducente quando temos programas com milhares de linhas de código – neste caso a melhor estratégia seria o uso de *breakpoints*.

2.9 *Double click* sobre a instrução '*sjmp*' na janela de *disassembly* e aparecerá uma marca vermelha na margem esquerda que indica a inserção de um *breakpoint*.



🌠 teste 🛛 - µVision	n3 - [Disasse	mbly]			
<u>F</u> ile <u>E</u> dit ⊻iew	Project Debu	ıg Fl <u>a</u> sh Peripherals <u>T</u> o	ools <u>S</u> VCS <u>W</u> ir	ndow <u>H</u> elp	
🏠 🚅 🗟 🎒	አ 🗈 📾	白白津津水	36 36 16	74	
않 🗉 🛛 🖓	0+ 10 1 0	♦ ₩ 03 🖗 🔊	🖤 😸 🔳 I	E 🚟 🖬 🎤	
Project Workspace	▲ X	COMODES	00	NOP	
Begister	Value	C:0x00F6	00	NOP	Darallal Dart 1
Bens	- Glub	C:OxOOF7	00	NOP	
- nogo	0×00	C:OxOOF8	00	NOP	Port I
	0x00	C:0x00F9	00	NOP	
r2	0x00	C:OxOOFA	00	NOP	Pine DuED FURTHER
r3	0x00	C:0x00FB	00	NOP	
r4	0x00	C:0x00FC	00	NOP	
-15	0x00	C:OxOOFD	00	NOP	
r6	0x00	C:OxOOFE	00	NOP	
17	0x00	C:UXUUFF	00	NOP	1 10 01
⊡ Sys		9: Sta:	rt: mov		pl,#UeUn ; make pits 5 tU / inputs
a	0xe0	10:	750050	MOM	D1(0-00) #XCC(0-E0)
ь	0x00	11: loo	739020	MOV	PI(0X90), #ACC(0XE0)
sp	0x07	12.	p. 110v		a,pi , iead ri
sp_max	0x07	C • 0v0103	F590	MOV	≥ P1(0v90)
aptr	0x0000	13:	2000	mov	R^{2} ,#5 :shift right 5x (acc >> 5)
FL 3	5.0x01	C:0x0105	7A05	MOV	R2.#0x05
sidies	0,0000	14: rot	lp: rr		a
± nsw	0x01	C:0x0107	03	RR	A
- poin	ono i	15:		djnz	r2,rotlp
		C+0×0108	DAFD	D INZ	P2 POTLP(C+0107)
		17.	DHID	anl	a #7 :mask input hits
		C:0x010A	5407	ANI	A_#0x07
		18:		mov	dptr.#table :get table address
		C:0x010C	900003	MOV	DPTR,#0x0003
		19:		move	a,@a+dptr ;get ith entry in table
		C:0x010F	93	MOVC	A_@A+DPTR
		21:		orl	a, #OeOh ;make high order bits = 1
		C:0x0110	44E0	ORL	A,#ACC(0xEO)
		22:		mov	pl,a ;output to pl
		C:0x0112	F590	MOV	P1(0x90),A
		23:		sjmp	loop ; repeat forever
		C:0x0114	SOED	SJMP	L00P(C:0103)
		C:0x0116	00	NOP	
		C:Ux0117	UU	NOP	

Se tentar executar o programa no modo contínuo (F5) este parará a execução quando atingir o *breakpoint* – neste caso a localização da seta amarela coincidiria com a localização do '*sjmp*'.

🕎 teste - µVision	3 - [Disasse	embly]			
Eile Edit View	Project Deb	ug Fl <u>a</u> sh Pe <u>r</u> ipherals j	<u>T</u> ools <u>S</u> VCS <u>W</u> i	ndow <u>H</u> elp	
Marce n	X Ba PA		1 % % %	634	
			0 /0 /0 /0		
RST 🗒 🛛 🔁	{} ¹ {} ⁺ * {}	◆ ¥¥ 0£ Q &	3 🎨 🧏 🔲	E 🛛 🖪 🗡	
Project Workspace	• ×	C:0x00F5	00	NOP	
Register	Value	C:0x00F6	00	NOP	Parallel Port 1
E Regs		C:UXUUF7	00	NOP	Port 1
rO	0x00	C:UXUUF8	00	NOP	P1: DvE1 7 Bits 0
n	UxUU	C.0x00F5 C.0x00F5	00	NOP	
12	0.00	C:0x00FB	00	NOP	
r4	0x00	C:OxOOFC	00	NOP	
6	0x00	C:OxOOFD	00	NOP	
r6	0x00	C:OxOOFE	00	NOP	
17	0x00	C:OxOOFF	00	NOP	
⊟— Sys		9: sta	art: mov		p1,#OeOh ; make bits 5 tO 7 inputs
a	Oxf1	10:	750050	MOV	D1/000) #X((/0E0)
ь	UxUU	11: lor	7350E0 20: MOX	MOV	a n1 · read P1
sp op. mou	0x07 0x07	12:	sp. mos		dypi y ioda ii
sp_max	0x07	C:0x0103	E590	MOV	A,P1(0x90)
PC \$	C:0x01	13:		mov	R2,#5 ;shift right 5x (acc >> 5)
states	28	C:0x0105	7A05	MOV	R2,#0x05
sec	0.0000	14: rot	lp: rr		a
÷ psw	0x01	C:UxU1U7	03	RR	A
		15:		djnz	r2,rot1p
		C·0v0108	DAFD	D INZ	02 00TT 0(C+0107)
		17:	DIND	anl	a.#7 :mask input bits
		C:0x010A	5407	ANL	A,#0x07
		18:		mov	dptr,#table ;get table address
		C:0x010C	900003	MOV	DPTR,#0x0003
		19:		move	a,@a+dptr ;get ith entry in table
		2U:	00	Nova	
		21.	90	move	A, WA+DFIK = #0c0b :make bigh order bits = 1
			44F0	ORI	A #ACC(NyFO)
		22:		mov	pl.a :output to pl
		C:0x0112	F590	MOV	P1(0x90),Å
		23:		sjmp	loop ; repeat forever
		C:0x0114	80ED	SJMP	LOOP(C:0103)
		C:0x0116	00	NOP	
		L C:0x0117	00	NOP	



Suponhamos que chegou a conclusão que a simulação revelou uma inconsistência entre o valor previsto e o pretendido. Imagine que a correcção necessária consistiria na execução de duas novas instruções:

antes da execução da instrução 'orl' na localização C:0110H. Como alteraria o código anterior?

2.10 "Clique" o *icon debug*, (a), para voltar ao modo de edição e adicione as duas instrução ao código fonte.



"Assemble" a aplicação como anteriormente indicado e volte ao modo de depuração. Abra agora ambas janelas para o porto 1 e para o porto 3. Em vez de executar o código passo-a-passo ou no modo contínuo com breakpoints, active o item de menu 'view/periodic window update'.





Neste modo o código é executado continuamente (F5) e periodicamente o estado dos portos (ou de qualquer outro periférico seleccionado) será actualizado. Experimente alterar o estado dos *bits* 5-7 do porto 1 e verifique o que ocorre.

Nota:

Repare que poderia extrair algumas informações úteis consultando o ficheiro '.lst'. Por exemplo, passe para o modo de edição e seleccione 'File/Open' e abra o ficheiro com extensão '.lst'.

	📅 teste - µVision3 - [C:\home\mic-8051\proj-1\teste.LST]									
Image: Source with a state of the	Eile Edit View Project Debug	Flash Peripherals Tools SVCS Wind	dow <u>H</u> elp							
Image:	🏠 😂 🖬 🕼 👗 🛍 🛍	22 倖存人%%%%	74	- 44	→ M	→ (@ @) 🔍 🔼 🖪 🖉 (
Point Workspace x	🗇 🕮 🆽 👗 🙀 🌾 Targe	et 1 💽 📥 🥊								
Target: System Syste	Project Workspace - x	A51 MACRO ASSEMBLER	TESTE					10/03/2006 12:15:03 PAGE 1		
Source Group 1 NACRO ASSERBLER ASI V8.00b OBJECT MODULE PLACED IN Veste.OBJ ASSERBLER INVOKED BY: C:\Program Files\keil\C51\BIN\ASI.EXE teste.aSI SET(SMALL) DEBUG EP Loc OBJ LINE 1 ; example for a p39-like problem 3 Cseg at 0000 020100 4 1jmp 0000 020100 5 cseg at 0 0000 020100 5 0 start: 0 0000 020100 5 cseg at 100h 0000 020100 5 start: mov a,p1 ; read P1 0100 7500E0 9 start: mov R2,#/5 ; shift right 5x (acc >> 5) 0100 11 loop: mov a,#/7 ; mask input bits 0100 15 mov a,#/7 ; mask input bits 0100 1000 mov c,aco.5 ; 0110 A2E5 1 mov c,aco.5 0110 21 mov c,aco.5 ; </th <th>- Target 1</th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>	- Target 1									
■ BRACRO ASSEMBLER ASI V8.000 ODJECT MOULE PLACED IN teste.05J ASSEMBLER INVOKED BY: C:\Program Files\Keil\C5\\BIN\A51.EXE teste.a51 SET(SMALL) DEBUG EP Loc OBJ LINE SOURCE	🖻 😋 Source Group 1									
OBJECT MOULE FLACED IN tests.037 ASSEMBLER HVOKED BY: C:\Program Files\Keil\CSi\BIN\ASI.EXE tests.a51 SET(SMALL) DEBUG EP LOC OBJ LINE SOURCE 1 ; example for a p39-like problem 2 Cseg at 0 0000 020100 4 1jmp start 0000 1010409 6 table: db 0,1,4,9,16,25,36,49 0007 10192431 7 7 6 Cseg at 100h 0100 7590E0 9 start: mov p1,#0eOh ; make bits 5 t0 7 inputs 0103 E590 10 loop: mov a,p1 ; read P1 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0107 03 14 rotlp: rr ajm r2,rotlp 0104 5407 17 an1 a,#7 ; mask input bits 0105 7A05 13 mov ayet adptr ; get time tury in table 0104 5407 17 an1 a,#7 ; zmask input bits 0105 7A05 20 mov c,acc.5 0104 255 21 mov c,acc.5 0105 730 2 mov p3.0,c 0110 A225 21 </th <th>teste.a51</th> <th>MACRO ASSEMBLER A51 V</th> <th>78.00b</th> <th></th> <th></th> <th></th> <th></th> <th></th>	teste.a51	MACRO ASSEMBLER A51 V	78.00b							
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$		OBJECT MODULE PLACED	IN teste.OBJ							
LOC OBJ LINE SOURCE		ASSEMBLER INVOKED BY:	: C:\Program 1	Files\Keil\	C21/BIN	AS1.EXE	teste.a51 SET(SM	IALL) DEBUG EP		
1 cc 000 1 ml SOUNCE 1 ; example for a p39-like problem 2 3 Cseg at 0 0000 020100 4 1jmp start 0000 710192431 7 8 cseg at 100h 0100 7590E0 9 start: mov p1,#0e0h ; make bits 5 t0 7 inputs 0100 7590E0 12 mov a,p1 ; read P1 0103 E590 11 loop; mov a,p1 ; read P1 0103 F590 12 mov R2,#5 ; shift right 5x (acc >> 5) 0103 E590 13 mov a,f7 ; mask input bits 0103 F590 15 djnz r2,rotlp 16 0104 S407 17 anl a,#7 ; mask input bits 0104 S407 17 anl a,#40tr ; get table address 0105 F33 19 mov c,acc.5		LOC ORI I	THE SOUD	CF						
1 ; example for a p39-like problem 0 0000 020100 4 1jmp 0003 00010109 6 table: db 0003 0001012431 7 0100 7500E0 9 start: mov 0103 ES90 11 loop: mov 0103 ES90 11 100 loop: mov 0103 ES90 11 103 ES90 13 11 loop: mov 11 loop: mov 11 loop: mov 11 loop: mov 12 rotlp: rr 13 mov R2,#5 ;shift right 5x (acc >> 5) 1010 A mov a,#2 jmexk input bits 1010 A mov a,#2 jmexk input bits 1010 A mov a,#0 jmexk input bits 1010 A		100 000 1	SIME SOUR	05						
3 cseg at 0 0000 020100 4 1jmp start 0003 00010409 6 table: db 0,1,4,9,16,25,36,49 0007 10192431 7 cseg at 100h 8 cseg at 100h 0100 7590ED 9 start: mov 0103 E590 11 loop: mov a,p1 10 11 loop: mov a,p1 0105 7A05 13 mov R2,#5 0105 7A05 13 mov 0108 DAFD 15 djnz r2,rotlp 1010 42E5 21 mov 0101 A2E5 21 mov 0110 A2E5 21 mov 0114 44ED 24 orl 0116 80E9 26 27 28 end 11.400 0116 80E9 26 27 and 0116 80E9 26 20 mov p1.a 2118 80E9 26 23 mov 0118 80E9 26 27 and 0121 14HEL LISTING 2			1 ; ex:	ample for a	p39-li	ke proble	m			
3 cseg at 0 0 0000 020100 4 1jmp start 0 0000 020100 6 table: db 0,1,4,9,16,25,36,49 0 0000 75000 9 start: mov cseg at 100h 0 0100 7590E0 9 start: mov a,1 ; read P1 0103 E590 11 loop: mov a,p1 ; read P1 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0105 7A05 13 mov adjut r2,rotlp a 0100 5407 16 mov a,#44ptr ; get table address a 0100 590003 18 mov c,acc.5 a 0100 593 19 mov c,acc.5 a a 0110 A2E55 21 mov c,acc.5 a a 0111 A2E5 21 mov c,acc.5 a a 0112 A2E5 23 mov p3.0,c a a a 0116 F590			2							
0000 020100 4 ljmp start 0003 00010409 6 table: db 0,1,4,9,16,25,36,49 0007 10192431 7			3		cseg	at	0			
0003 00010409 6 table: db 0,1,4,9,16,25,36,49 0007 10192431 7 cseg at 100h 0100 7590E0 9 start: mov a,p1 ; make bits 5 t0 7 inputs 0103 E590 11 loop: mov a,p1 ; read P1 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0107 03 14 rotlp: rr a djnz r2,rotlp 0106 5407 17 anl a,#7 ;mask input bits 0106 900003 18 mov djnz r2,rotlp 0106 93 20 mov c,acc.5 jet table address 0101 A2E5 21 mov c,acc.5 0112 A2E5 21 mov r2,acd.5 0114 44E0 23 orl a, #0e0h ;make high order bits = 1 0116 F590 25 mov p1,a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 28 0131 MACFO ASSEMBLER TEST 10/03/2006 12:15:03 PAGE 2		0000 020100	4		ljmp	start				
0003 00014099 6 table: db 0,1,4,9,16,25,36,49 0007 10192431 7			5							
0007 10192431 7 cseg at 100h 0100 7550E0 9 start: mov pl,#0e0h ; make bits 5 t0 7 inputs 0100 7550E0 10 10 a,pl ; read Pl 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0107 03 14 rolp: rr a 0108 DAFD 15 djnz r2,rotlp 100 100 500003 18 mov a,#7 0100 793 19 mov a,#4dptr ;get table address 0100 225 21 mov c,acc.5 0110 A2EFS 21 mov c,acc.5 0112 92B0 23 mov pl.4 0114 44E0 24 orl a, #0e0h ;make high order bits = 1 0116 60E9 26 sjmp loop ; repeat forever 2 27 28 end 10/03/2006 12:15:03 PAGE 2 0XHOL TABLE LISTING TABLE LISTING 10/03/2006 12:15:03 PAGE 2		0003 00010409	6 tabl	e: db	0,1,4,	9,16,25,3	6,49			
8 cseg at 100h 0100 7590E0 9 start; mov pl,#OeOh ; make bits 5 t0 7 inputs 0103 E590 11 loop: mov a,pl ; read Pl 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0107 03 14 rotlp: rr a 0106 DAFP 15 djnz r2,rotlp 100 D105 18 mov dptr,#table ;get table address 0107 93 19 mov c,acc.5 c.acc.5 0102 2820 22 mov p3.0,c 0114 44E0 24 orl a, #GeOh ;make high order bits = 1 0116 F590 25 mov p1,a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 27 28 end 10/03/2006 12:15:03 PAGE 2		0007 10192431	-							
0100 7590E0 9 start: mov pl,#0e0h ; make bits 5 t0 7 inputs 0103 E590 11 loop: mov a,pl ; read P1 0105 7A055 13 mov R2,#5 ; shift right 5x (acc >> 5) 0107 03 14 rolp: rr a 0108 DAFD 15 djnz r2,rolp 0108 DAFO 16 a ,#7 ;mask input bits 0100 S0003 18 mov dptr,#table ;get table address 0107 93 19 mov c,acc.5 c 0110 A2E5 21 mov c,acc.5 c 0110 A2E5 21 mov c,acc.5 c 0110 A2E5 21 mov pl,#0e0h ;make high order bits = 1 0116 A2E5 24 orl a, #0e0h ;make high order bits = 1 0116 F530 25 mov pl,a ;output to pl 0118 B0E9 26 sjmp lo			,		0000		1001			
10 10 <t< th=""><th></th><th>0100 759080</th><th>9 stari</th><th>t: mov</th><th>cseg</th><th>n1.#0e0</th><th>h : make</th><th>bits 5 t0 7 innuts</th></t<>		0100 759080	9 stari	t: mov	cseg	n1.#0e0	h : make	bits 5 t0 7 innuts		
0103 E590 11 loop: mov a,p1 ; read P1 0105 7A05 13 mov R2,#5 ; shift right 5x (acc >> 5) 0107 03 14 rotlp: rr a 0108 DAFP 15 djnz r2,rotlp 16 nov a,#7 ;mask input bits 0106 500003 18 mov dptr,#table ;get table address 0107 93 19 mov c,acc.5 c 0110 A2E5 21 mov c,acc.5 c 0110 A2E5 21 mov p3.0,c c 23 23 mov p1,a ;output to p1 0116 FS90 25 mov p1,a ;output to p1 0116 FS90 26 sjmp loop ; repeat forever 28 28 end 27 28 end 28 24 27 20/03/2006 12:15:03 PAGE 2			10			F-,	- ,			
12 mov R2,#5 ;shift right 5x (acc >> 5) 0107 03 14 rotlp: rr a 0108 DAFD 15 djnz r2,rotlp 16 ani a,#7 ;mask input bits 0106 5407 17 ani a,#7 ;mask input bits 0106 500003 18 mov djnz r2,rotlp 0107 93 19 mov a,@a+dptr ;get table address 0110 A2ES 21 mov c,acc.5 c 0112 92B0 22 mov p3.0,c 23 0114 44EO 24 orl a, #0eOh ;make high order bits = 1 0116 505 25 mov p1.a ;output to p1 0116 60E9 26 sjmp loop ; repeat forever 28 0A51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2		0103 E590	11 loop	: mov		a,p1	; read	P1		
0105 7A05 13 mov F2,#5 ;shiftright 5x (acc >> 5) 0107 03 14 rotlp: rr a 0108 DAFD 15 djnz r2,rotlp 16 16 0105 90003 18 mov dptr,#table ;get table address 0100 90003 18 mov dptr,#table ;get table address 0101 A2E5 21 mov c,acc.5 0110 A2E5 21 mov c,acc.5 0112 92B0 22 mov p3.0,c 23 23 23 0114 44E0 24 orl a, #0e0h ;make high order bits = 1 0116 F590 25 mov p1.a ;output to p1 0118 80E9 26 sjmp <loop< td=""> ; repeat forever 2 27 28 end 28 2 10/03/2006 12:15:03 PAGE 2 0A51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2</loop<>			12							
0107 03 14 rotlp: rr a 0108 DAFD 16 djmz r2,rotlp 16 100 r2,rotlp 16 anl a,#7 ;mask input bits 0100 900003 18 mov dptr,#table ;get table address 0107 93 19 mov a,8a+dptr ;get ith entry in table 20 20 mov c,acc.5 0110 A2ES 21 mov c,acc.5 0112 92B0 22 mov p3.0,c 23 114 44E0 24 orl a, #0e0h 0116 F590 25 mov p1,a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 27 28 end 27 28 end 27 28		0105 7805	13		mov		R2,#5	;shift right 5x (acc >> 5)		
0108 DAFD 15 djnz r2,rotlp 16 16 100 5407 17 anl a,#7 0102 900003 18 mov dptr,#table 0107 93 19 mov a,@a+dptr 20 20 100 25 0110 A2E5 21 mov c,acc.5 0112 92B0 22 mov p3.0,c 23 23 23 0114 44E0 24 orl a, #0e0h 0118 80E9 26 sjmp loop 27 28 end 28 01A51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2		0107 03	14 rotl	p: rr		a				
16 an1 a,#7 ;mask input bits 010A 5407 17 an1 a,#7 ;mask input bits 010C 900003 18 mov dptr,#table ;get table address 010F 93 19 mov a,@a+dptr ;get table address 010F 93 20 mov c,acc.5 0110 A2E5 21 mov c,acc.5 0112 92B0 22 mov p3.0,c 23 23 23 23 0114 44E0 24 orl a, #0e0h ;make high order bits = 1 0116 F590 25 mov p1.a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 28 27 28 end 28 28 24 0A51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2		0108 DAFD	15		djnz	r2,rotl	p			
0100.3900 17 ani a, μ,		0103 5407	10		onl		o #7	meet input hite		
OIOF 93 15 move d, @a+dptr ; get ith entry in table 0IOF 93 19 move d, @a+dptr ; get ith entry in table 20 20 move c, acc.5 0IIO A2E5 21 move c, acc.5 0I12 92B0 22 move p3.0,c 20 23 0I14 44E0 24 orl a, #0e0h ; make high order bits = 1 0I16 F590 25 move p1,a ; output to p1 0I18 80E9 26 sjmp loop ; repeat forever 27 28 end DA51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2		0100 900003	18		mon		datr #teble	get table address		
Oli 0 20 mov c,acc.5 0110 A2E5 21 mov c,acc.5 0112 92B0 22 mov p3.0,c 23 23 23 24 orl a, #0eOh ;make high order bits = 1 0116 F590 25 mov p1.a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 27 28 end 28 DASI MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2		010F 93	19		move	a.0a+dn	tr : det it	h entry in table		
0110 A2ES 21 mov c,acc.5 0112 92B0 22 mov p3.0,c 23 23 23 0114 44E0 24 orl a, #0e0h 0116 FS90 25 mov p1.4 0116 60E9 26 sjmp loop 27 28 end 28 DA51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2			20			, ,	, , ,			
0112 9280 22 mov p3.0,c 23 0114 44E0 24 orl a, #0e0h ;make high order bits = 1 0116 F590 25 mov p1,a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 27 28 end 10/03/2006 12:15:03 PAGE 2 DA51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2		0110 A2E5	21		mov		c,acc.5			
23 0114 44E0 24 orl a, #0e0h ;make high order bits = 1 0116 F590 25 mov p1,a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 27 28 end DASI MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2 3YMR0L TABLE LISTING		0112 9280	22		mov		р3.0,с			
0114 44E0 24 orl a, #OeOh ;make high order bits = 1 0116 F590 25 mov p1,a ;output to p1 0118 80E9 26 sjmp loop ; repeat forever 27 28 end DA51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2 SYMBOL TABLE LISTING			23							
Ullo FS9U 25 mov pl,a ;output to pl 0118 80E9 26 sjmp loop ; repeat forever 27 28 end DA51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2 SYMROL TABLE LISTING		0114 44E0	24		orl		a, #0e0h	;make high order bits = 1		
Clie GUES 20 Sjmp 100p ; repeat forever 27 28 end DASI MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2 SYMBOL TABLE LISTING		U116 F590	25		mov	1	p1,a	;output to pl		
28 end DAS1 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2 SYMROL TABLE LISTING		OITO OOFA	20		ajmp	100b	; repea	IC TOTEAET		
DA51 MACRO ASSEMBLER TESTE 10/03/2006 12:15:03 PAGE 2 SYMROL TABLE LISTING			28		end					
SYNROL TABLE LISTING		DA51 MACRO ASSEMBLER	TESTE					10/03/2006 12:15:03 PAGE 2		
		SYMBOL TABLE LISTING								

