Escola de Engenharia Elétrica, Mecânica e de Computação
Universidade Federal de Goiás

Laboratório de Microprocessadores e Microcontroladores

Experimento 1

Introdução ao Microcontrolador 8051

Alunos: _____ Matrícula: _____

_____ _____

_____ _____

_____ _____

Prof. Dr. José Wilson Lima Nerys

Goiânia, 1º semestre de 2019

SUMÁRIO

1 Introdução ao Microcontrolador 8051

1.1 Características Gerais do 8051

A pinagem do microcontrolador básico de 40 pinos da família 8051 é mostrada na Fig. 1.1, que mostra também o componente de 20 pinos.

O componente básico de 40 pinos contém 2 contadores/temporizadores, 4 portas paralelas de 8 bits, 2 fontes externas de interrupção e 3 internas, uma porta serial com um canal de entrada e outro de saída, memória RAM e memória ROM. O componente de 20 pinos diferencia-se, basicamente, por ter apenas duas portas de entrada/saída (portas P1 e P3).

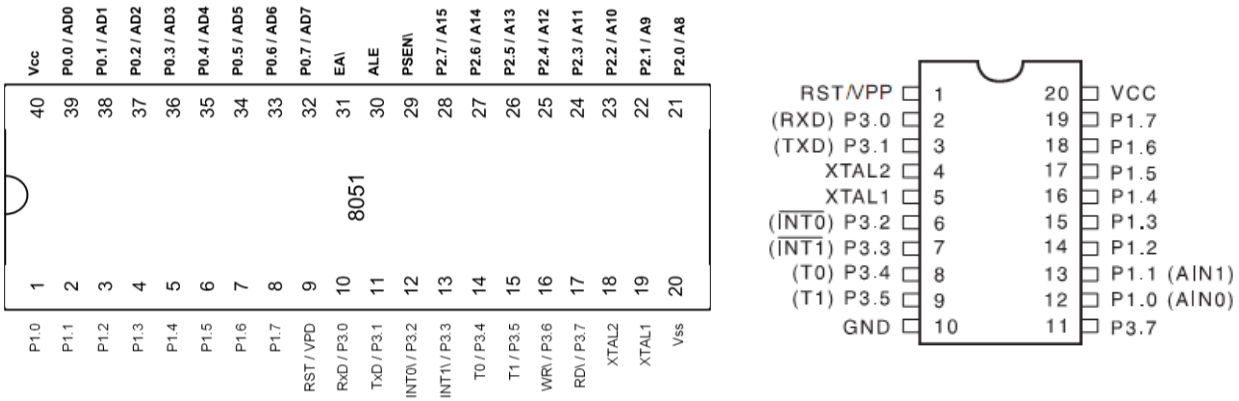


Fig. 1.1 – Pinagem dos microcontroladores de 40 pinos e 20 pinos da família 8051.

A Fig. 1.2 mostra o circuito mínimo necessário para acionamento de um LED através do pino 0 da porta P2. O driver constituído de um transistor e um dois resistores é necessário para o acionamento do LED através do microcontrolador de 40 pinos porque a capacidade de corrente desse componente é muito pequena. De acordo com o datasheet do componente AT89S52, a capacidade de corrente por pino de cada porta é de 10 mA e por porta de 8 pinos é de 15 mA para as portas P1, P2 e P3 e 26 mA para a porta P0.

Por outro lado, a capacidade de corrente do componente de 20 pinos é maior. De acordo com o datasheet do componente AT89C2051, a capacidade por pino é de 20 mA e a capacidade total para todos pinos é de 80 mA. Assim, o LED pode ser acionado diretamente, apenas com o resistor de 330 ohms para limitar a corrente, como mostrado na Fig. 1.2.

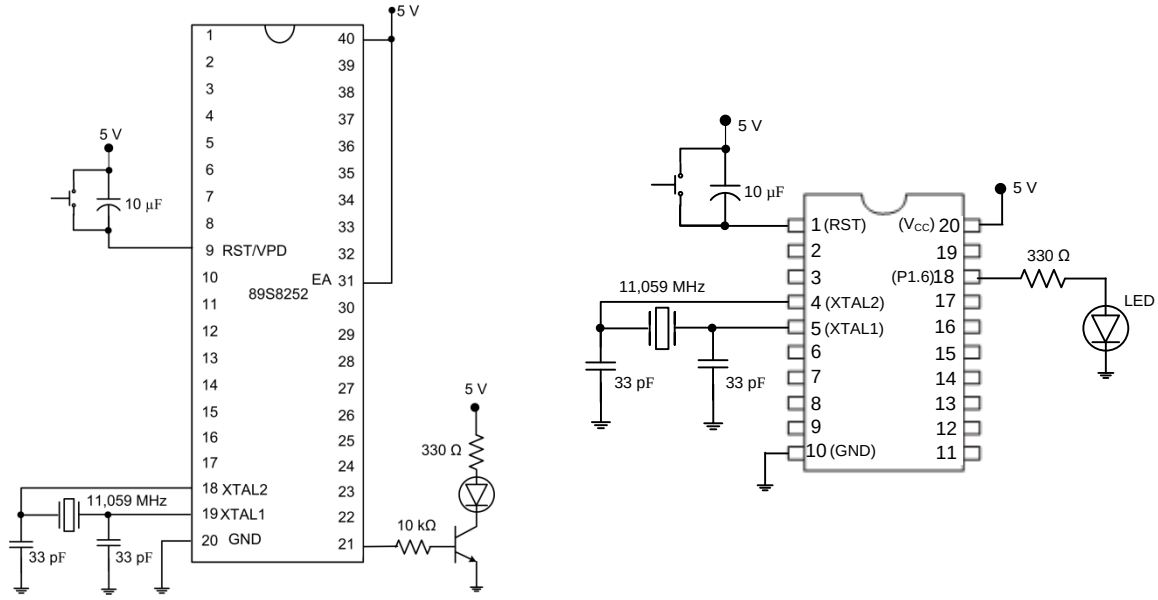


Fig. 1.2 – Sistema mínimo para acionamento de um LED

A Fig. 1.3 mostra o diagrama de blocos de um microcontrolador básico da família 8051.

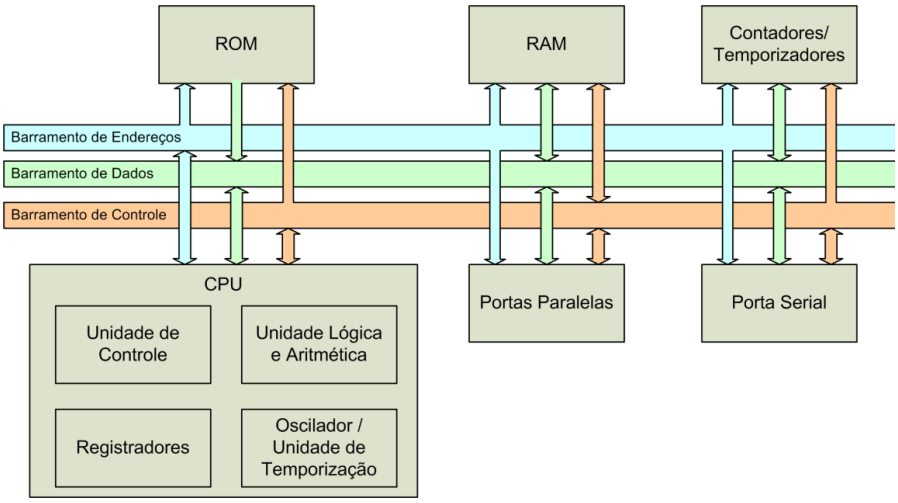


Fig. 1.3 – Diagrama de blocos de microcontrolador básico da família 8051

Destaca-se a memória RAM, Fig. 1.4, onde estão presentes os registradores utilizados nas instruções e onde há uma área reservada para os Registradores Especiais, tais como os registradores das portas P0 a P3 e os registradores TMOD e IE, de configuração dos temporizadores e das interrupções, respectivamente.

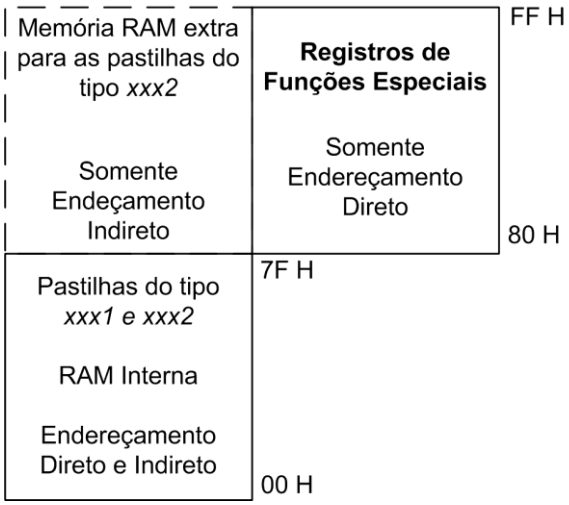
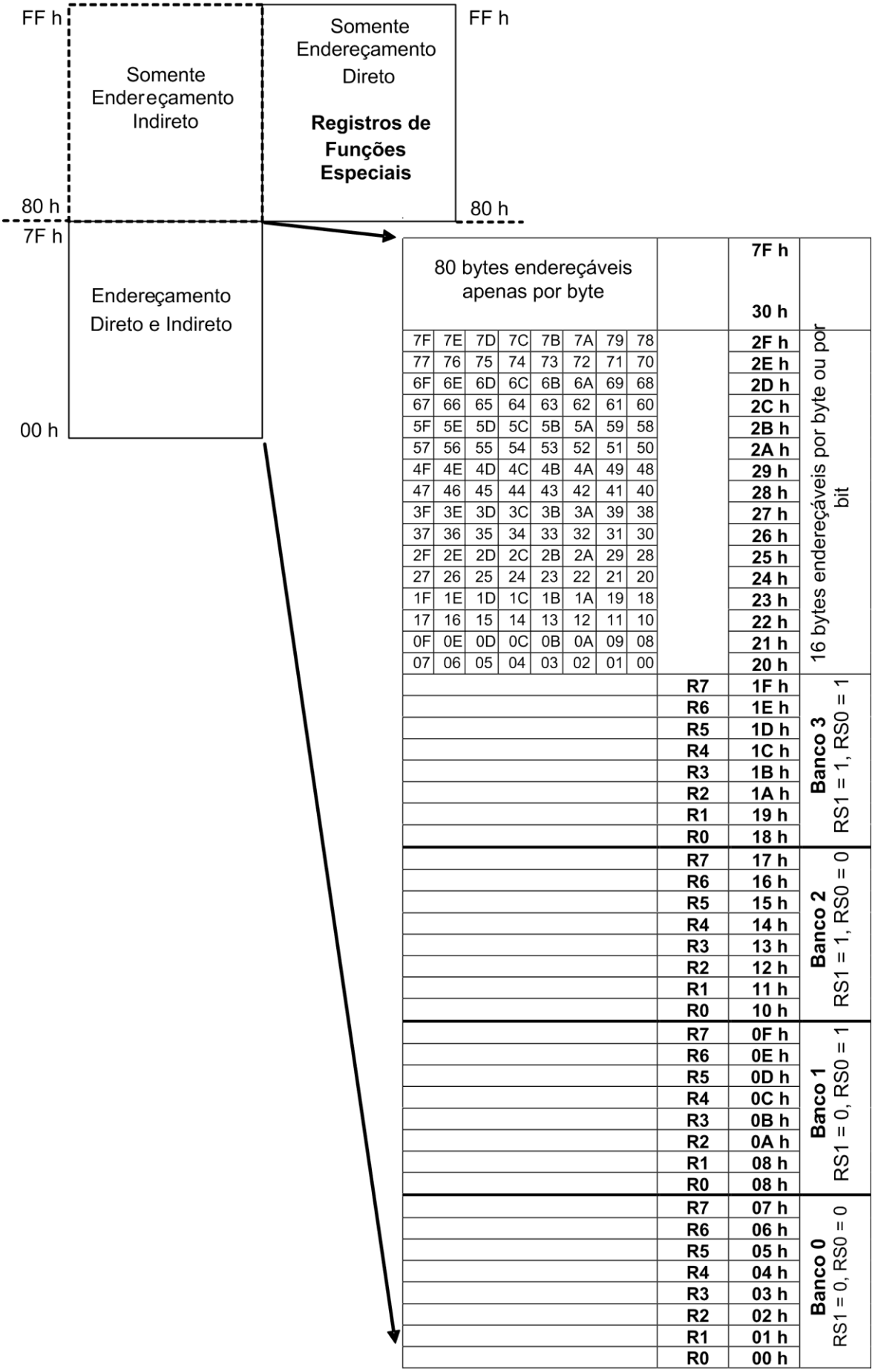


Fig. 1.4 – Memórias RAM interna

A Fig. 1.5 detalha a parte baixa da memória RAM, onde estão presentes 4 conjuntos de 8 registradores cada um, uma região de memória que pode ser acessada por bit e por byte e uma região de memória que pode ser acessada apenas por byte.



80 bytes endereçáveis apenas por byte									7F h – 30 h
7F	7E	7D	7C	7B	7A	79	78		2F h
77	76	75	74	73	72	71	70		2E h
6F	6E	6D	6C	6B	6A	69	68		2D h
67	66	65	64	63	62	61	60		2C h
5F	5E	5D	5C	5B	5A	59	58		2B h
57	56	55	54	53	52	51	50		2A h
4F	4E	4D	4C	4B	4A	49	48		29 h
47	46	45	44	43	42	41	40		28 h
3F	3E	3D	3C	3B	3A	39	38		27 h
37	36	35	34	33	32	31	30		26 h
2F	2E	2D	2C	2B	2A	29	28		25 h
27	26	25	24	23	22	21	20		24 h
1F	1E	1D	1C	1B	1A	19	18		23 h
17	16	15	14	13	12	11	10		22 h
0F	0E	0D	0C	0B	0A	09	08		21 h
07	06	05	04	03	02	01	00		20 h
								R7	1F h
								R6	1E h
								R5	1D h
								R4	1C h
								R3	1B h
								R2	1A h
								R1	19 h
								R0	18 h
								R7	17 h
								R6	16 h
								R5	15 h
								R4	14 h
								R3	13 h
								R2	12 h
								R1	11 h
								R0	10 h
								R7	0F h
								R6	0E h
								R5	0D h
								R4	0C h
								R3	0B h
								R2	0A h
								R1	08 h
								R0	08 h
								R7	07 h
								R6	06 h
								R5	05 h
								R4	04 h
								R3	03 h
								R2	02 h
								R1	01 h
								R0	00 h

Fig. 1.5 – Detalhes da Parte baixa da memória RAM interna

1.2 Os Registradores de Funções Especiais

A Tabela 1 mostra os principais **Registradores Especiais**, que ficam localizados na região de **80h a FF h** da memória RAM. Os registradores dessa região, com endereços de final **0** ou **8**, são endereçáveis por byte ou por bit. Os demais, apenas por byte.

Deve ser enfatizado que os registradores especiais ocupam os endereços de **80h** a **FFh**, que coincide com os 128 bytes superiores da RAM interna dos microcontroladores xxx2. A diferença entre o acesso aos Registradores especiais e a parte superior da RAM interna é o tipo de endereçamento. Os ***registros especiais*** são acessados sempre por ***endereçamento direto***, enquanto a parte superior da *RAM interna* é acessada somente por *endereçamento indireto*.

Tabela 1: Principais Registradores Especiais

Registrador	Mnemônico	Endereço	Endereços individuais dos Bits e denominações de alguns bits							
Latch da Porta 0	P0	80 H	87	86	85	84	83	82	81	80
Apontador de Pilha	SP	81 H								
Apontador de Dados	DPTR	82H – 83H								
LSB do Apontador de Dados	DPL	82 H								
MSB do Apontador de Dados	DPH	83 H								
Controle de Energia	PCON	87 H	SMOD							
Controle do Contador/Temporizador	TCON	88 H	8F	8E	8D	8C	8B	8A	89	88
			TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Controle do Modo do Temporizador/ Contador	TMOD	89 H	G1	C/T1	M11	M01	G0	C/T0	M10	M00
LSB do Temporizador/Contador 0	TL0	8A H								
LSB do Temporizador/Contador 1	TL1	8B H								
MSB do Temporizador/Contador 0	TH0	8C H								
MSB do Temporizador/Contador 1	TH1	8D H								
Latch da Porta 1	P1	90 H	97	96	95	94	93	92	91	90
Controle da Porta Serial	SCON	98 H	9F	9E	9D	9C	9B	9A	99	98
			SM1	SM2	SM3	REN	TB8	RB8	TI	RI
Porta de Dados Seriais	SBUF	99 H								
Latch da Porta 2	P2	A0 H	A7	A6	A5	A4	A3	A2	A1	A0
Habilitador de Interrupção	IE	A8 H	AF	AE	AD	AC	AB	AA	A9	A8
			EA			ES	ET1	EX1	ET0	EX0
Latch da Porta 3	P3	B0 H	B7	B6	B5	B4	B3	B2	B1	B0
Controle de Prioridade da Interrup.	IP	B8 H	BF	BE	BD	BC	BB	BA	B9	B8
						PS	PT1	PX1	PT0	PX0
Registrador de Estado do Programa	PSW	D0 H	D7	D6	D5	D4	D3	D2	D1	D0
			CY	AC	F0	RS1	RS0	OV		P
Acumulador	ACC ou A	E0 H	E7	E6	E5	E4	E3	E2	E1	E0
Registrador B	B	F0 H	F7	F6	F5	F4	F3	F2	F1	F0

O **PSW** (**P**rogram **S**tatus **W**ord) é o registrador especial que contém as Flags e também os bits RS1 e RS0, usados para selecionar o banco de registradores (ver Tabela 2). Este registrador é endereçável por bit. As flags do microcontrolador 8051 são: flag de carry (CY), flag auxiliar de carry (AC), flag de uso geral (F0), flag de overflow (OV) e flag de paridade (P).

PSW	Bit 7	Bit 6	Bit 5	**Bit 4**	**Bit 3**	Bit 2	Bit 1	Bit 0
	CY	AC	F0	**RS1**	**RS0**	0V		P

Tabela 2 – Seleção do banco de registradores

RS1	RS0	Banco Selecionado
0	0	0
0	1	1
1	0	2
1	1	3

A Tabela 3 apresenta os valores iniciais dos registradores especiais após o Reset. Observar que as portas de entrada/saída assumem nível lógico alto e o apontador de pilha assume o valor 07h.

Tabela 3 – Valores dos registradores especiais após o Reset

Registro	Valor	Registro	Valor
PC	0000h	TCON	00h
A	00h	TH0	00h
B	00h	TL0	00h
PSW	00h	TH1	00h
SP	07h	TL1	00h
DPTR	0000h	SCON	00h
P0 - P3	FFh	SBUF	Indeterminado
IP	xxx00000b	PCON(NMOS)	0xxxxxxxb
IE	0xx00000b	PCON(CMOS)	0xxx0000b
TMOD	00h		

1.3 Instruções Básicas Gerais do 8051

As instruções do 8051 podem ser digitadas em maiúsculas ou minúsculas. A seguir são mostradas algumas dessas instruções, com exemplos. O símbolo "#" é necessário para diferenciar dado de registrador. Os dados seguidos de "H" ou "h" estão no sistema hexadecimal; os dados seguidos de "B" ou "b" estão em binário e os dados sem nenhuma indicação estão no sistema decimal. Os termos "DIRETO", "dir'" e "dir2" significam referência ao endereço do registrador, ao invés de seu nome. O termo "REG" corresponde a Registrador, podendo ser de R0 a R7.

Instrução	**Descrição e exemplos**
MOV A,#DADO	Carrega o acumulador com o valor de "dado". MOV A,#25 → Carrega acumulador com valor decimal 25 (19 hexadecimal) MOV A,#15H → Carrega acumulador com valor hexadecimal 15H MOV A,#01011001b → Carrega acumulador com o binário equivalente a 59H
MOV A,DIRETO	Copia no acumulador o conteúdo do registrador cujo endereço é "direto". MOV A,15H → Copia no acumulador o conteúdo do registrador R5 (15H), do banco 2.
MOV A,REG	Copia no acumulador o conteúdo do registrador "reg", sendo reg = R0, R1, ..., R7, do banco de registradores que estiver ativo. MOV A,R6 → Copia no acumulador o conteúdo do registrador R6.
MOV dir2,dir1	Copia no registrador cujo endereço é "dir2" o conteúdo do registrador cujo endereço é "dir1". MOV 02H,05H → Copia em R2 (02H) o conteúdo do registrador R5 (05).

Instrução	Descrição e exemplos
MOV R0,#20H MOV @R0,#55H	Carrega registrador R0 com valor 20h Copia o valor 55h na posição apontada pelo registrador R0, ou seja, endereço 20H, que é a primeira posição acima do banco de registradores.
MOV DPTR,#200H MOVC A,@A+DPTR	Carrega registrador de 16 bits "dptr" com valor 200H Carrega acumulador com o conteúdo da posição apontada por "a + dptr". Se, por exemplo, A = 04H, então carrega acumulador com o conteúdo da posição 204H.
MOV DPTR,#200H MOVX @DPTR,A	Carrega registrador de 16 bits "dptr" com valor 200H Envia o conteúdo do acumulador para a posição externa 200H, apontada pelo DPTR
ADD A,REG	Adiciona o conteúdo do registrador "reg" ao conteúdo do acumulador. ADD A,R1 → Se A = 07 H e R1 = 03 H, então, após a instrução, a = 0AH.
ADD A,DIRETO	Adiciona o conteúdo do registrador de endereço "direto" ao conteúdo do acumulador: A = A + (direto) ADD A,10H → Se A = 07 H e 10H = 03 H, então, após a instrução, A = 0AH.
ADD A,#DADO	Adiciona ao conteúdo do acumulador o valor "dado": A = A + dado ADD A,#04h → Se a = 07 H, então, após a instrução, A = 0BH.
ADD A,@Rn	Adiciona ao conteúdo do acumulador o conteúdo da posição apontada por Rn. A = A + ((Rn)). MOV R0,#20h ADD A,@R0 → Se A = 07 H e registrador 20H = #03H, então, após a instrução, A = 0AH.
SUBB A,#DADO	Subtrai o conteúdo do acumulador do "DADO". A = A – DADO. SUBB A,#05H → Se A = 07 H, então, após a instrução, A = 02 H.
RL A	Rotaciona o conteúdo do acumulador para a esquerda (rotate left). Por exemplo, se originalmente A= 21 H (0010 0001b), após a instrução, tem-se: A = 42 H (0100 0010b).
RR A	Rotaciona o conteúdo do acumulador para a direita (rotate right). Por exemplo, se originalmente A= 8C H (1000 1100b), após a instrução, tem: A = 46 H (0100 0110b).
INC REG	Incrementa conteúdo do registrador "reg". Por exemplo, se R1 = 05H, então *INC R1* resulta em R1 = 06 H.
DEC REG	Decrementa conteúdo do registrador "reg". Por exemplo, se R2 = 0B H, então *DEC R2* resulta em R2 = 0A H.
CPL A	Complementa o conteúdo do acumulador. Por exemplo, se originalmente, A = 55 H, então, após a instrução, A = AA H.
SWAP A	Faz a troca dos nibbles do acumulador, ou seja, o nibble mais significativo passa a ocupar os quatro primeiros bits do acumulador e o nibble menos significativo passa a ocupar os quatro últimos bits. Por exemplo, se originalmente, A = 35 H, após a instrução, A = 53 H.
DA A	Faz o ajuste decimal do acumulador. Adiciona "6" ao dígito que esteja no intervalo de A a F. Por exemplo, se originalmente A = 7A H, após a instrução torna-se A = 80 H.
MUL AB	Multiplica o conteúdo de A pelo conteúdo de B. O resultado está em B A. O resultado da multiplicação é um número de 16 bits, por isso precisa de dois registradores para o resultado. MUL AB → se A = 25 H e B = 30 H, após a instrução, tem-se: B = 06 H e A = F0 H, pois o resultado da multiplicação é: 6F0 H

Instrução	Descrição e exemplos
DIV AB	Divide o conteúdo de A pelo conteúdo de B. A recebe o quociente e B o resto. DIV AB → se A = CA H (202) e B = 19 H (25), após a instrução, tem-se: A = 08 H e B = 02, pois a divisão em decimal (202/25) resulta em quociente 8 e resto 2.
ANL A,#DADO	Faz uma operação AND entre acumulador e DADO. A = A (AND) DADO. ANL A,#0FH → se originalmente A = 35 H, após a instrução torna-se: A = 05H.
ORL A,#DADO	Faz uma operação OR entre o acumulador e DADO. A = A (OR) DADO. ORL A,#20H → se originalmente A = 07 H, após a instrução torna-se: A = 27 H.

1.4 Instruções de Comparação, Decisão e de Desvio

As instruções desta seção são de desvio incondicional e desvio que depende do estado de flags.

Instrução	Descrição e exemplos
SJMP DESVIO	Desvio incondicional curto (Short Jump) relativo. Pula até 127 bytes para a frente e até 128 bytes para trás.
AJMP DESVIO	Instrução de desvio para distâncias correspondentes a até 2048 bytes. Endereço de 11 bits.
LJMP DESVIO	Desvio incondicional longo, para qualquer posição da memória de programa. Endereço de 16 bits.
JNZ DESVIO	Instrução de desvio condicional: Jump IF Not Zero. Pula para "desvio" se a operação anterior não resultar em zero. Verifica automaticamente a flag de zero.
LCALL SUBROT	Chamada de subrotina. Desvia para o endereço onde a subrotina está localizada. Ao encontrar a instrução RET, retorna para a instrução que vem logo após a chamada de subrotina.
JC DESVIO	Desvio condicional para a posição indicada por "desvio". Desvia se a flag de CARRY estiver setada.
JNC DESVIO	Desvio condicional para a posição indicada por "desvio". Desvia se a flag de CARRY não estiver setada.
DJNZ REG,DESVIO	Decrementa registrador "reg" e pula para a posição "desvio" se o resultado não for zero. É uma combinação das instruções "DEC" e "JNZ" do microprocessador 8085. MOV R5,#10 V1: DJNZ R5,V1 → O registrador R5 é decrementado até tornar-se zero
CJNE A,#DADO,V1	Compara conteúdo do acumulador com "dado" e pula para a posição "V1" se não forem iguais. MOV A,#00H V1: INC A CJNE A,#20H,V1 → Compara o conteúdo de A com 20 hexadecimal e, caso não seja igual pula para V1 para incrementar A. Quando for igual, pula para a próxima linha.

A diferença entre LJMP e SJMP é que a primeira instrução se refere a um endereço de 16 bits e é codificada em 3 bytes: o opcode e os dois bytes de endereço. A instrução SJMP é codificada em 2 bytes sendo o segundo byte o valor que deve ser adicionado à posição atual do apontador de programa **PC**, para determinar o endereço de desvio. O exemplo a seguir mostra um programa e sua codificação, com as instruções SJMP e LJMP. No programa mostrado o código de **SJMP V1** é **8009**, onde **80H** é o opcode da instrução e **09H** é o valor que deve ser adicionado ao contador de programa **PC** para indicar a próxima instrução a ser executada. Após a execução de **SJMP V1** o valor de **PC** é **0037H**. Adicionando 09H chega-se a **0040H**, endereço da instrução **ADD A,#53H**.

O código da instrução **LJMP INICIO** é **020030**, onde **02H** é o opcode da instrução e **0030H** é o endereço de desvio, ou seja, a posição de início do programa, para execução da instrução MOV A,#35H.

Endereço	Codificação	Rótulo	Mnemônico
			ORG 00H
0000	020030		LJMP INICIO
			ORG 30H
0030	7435	INICIO:	MOV A,#35H
0032	75F045		MOV B,#45H
0035	8009		SJMP V1
0037			
			ORG 40H
0040	2453	V1:	ADD A,#53H
0042	020030		LJMP INICIO
			END

1.5 Operações com bit

As instruções mostradas a seguir são algumas das instruções usadas nas operações com bit, ao invés de byte. O bit pode ser de um registrador especial (daqueles que permitem controle individual por bit) ou da região da memória RAM que vai do endereço 20H até 2FH.

Instrução	Descrição e exemplos
JB BIT,DESVIO	Desvia para a posição "desvio", caso o "bit" esteja setado. JB LIGADO,DESLIGA → Se o bit *ligado = 1*, então o programa desvia para a posição "deliga".
JNB BIT,DESVIO	Desvia para "desvio", caso o "bit" NÃO esteja setado. JNB LIGADO,LIGA → Se o bit *ligado = 0*, então o programa desvia para a posição "liga".
SETB BIT	Seta o "bit". SETB LIGADO → Torna o bit "ligado" igual a 1.
CLR BIT	Limpa o "bit" CLR LIGADO → Torna o bit "ligado" igual a zero

1.6 Diretivas de Programação

Durante a programação em assembly, são necessárias algumas informações para o compilador. Essas informações não são compiladas, mas apenas informam sobre variáveis, sobre posicionamento na memória e sobre dados. As principais diretivas são dadas a seguir:

org *endereço* → Informa ao compilador o endereço onde deve ser armazenada a próxima instrução.

Exemplo:

org 30 H

mov sp,#2Fh → Esta instrução será armazenada na posição 30 H da memória ROM.

variável **equ** *ender. reg.* → informa ao compilador que a "variável" *equivale* ao registrador cujo endereço é "*ender. reg*".

Exemplo:

velocidade equ 05H → Esta diretiva diz ao compilador que as operações com a variável "velocidade" equivalem às operações com o registrador R5 do banco 0 (endereço do registrador: 05 H). Por exemplo: mov velocidade,#52H equivale à instrução mov R5,#52H.

variável **bit** *ender. bit* → informa ao compilador que a "variável" é do tipo bit e será armazenada no endereço dado por "*ender.bit*".

Exemplo:

sentido bit 00H → Esta diretiva diz ao compilador que a variável "sentido" é do tipo bit e será armazenada no endereço 00H da região acima dos bancos de registradores. O endereço do **bit 00H** corresponde ao primeiro endereço dessa região, ou seja, posição **20.0H**.

db *byte* → Esta diretiva diz ao compilador que o byte a seguir é um dado e não uma instrução.

Exemplo:

db 45H → O valor 45H é tratado como um dado, não como uma instrução.

1.7 O Simulador Digital e o Kit Didático

A edição, compilação e simulação dos programas desenvolvidos para o 8051 tem sido realizada utilizando o simulador MCU 8051, cuja tela inicial é mostrada na Fig. 1.6. A compilação gera arquivos com as extensões ".hex", e ".lst", além de outras extensões. O arquivo ".hex" é usado simulação usando o Proteus e na gravação do microcontrolador usado no Kit Didático.

O Kit Didático complementa os simuladores. Ele consiste de uma plataforma mínima com um microcontrolador, um teclado, um conjunto de 8 LEDs, um módulo de comunicação serial e conectores dando acesso às portas P0, P1, P2 e P3, de modo que módulos diversos podem ser conectados. Já estão prontos os seguintes módulos: módulo de motor de passo; módulo de motor de corrente contínua; módulo de relés e sensor de presença; módulo de display LCD e módulo de conversores AD e DA. A Figura 1.7 mostra um diagrama do kit.

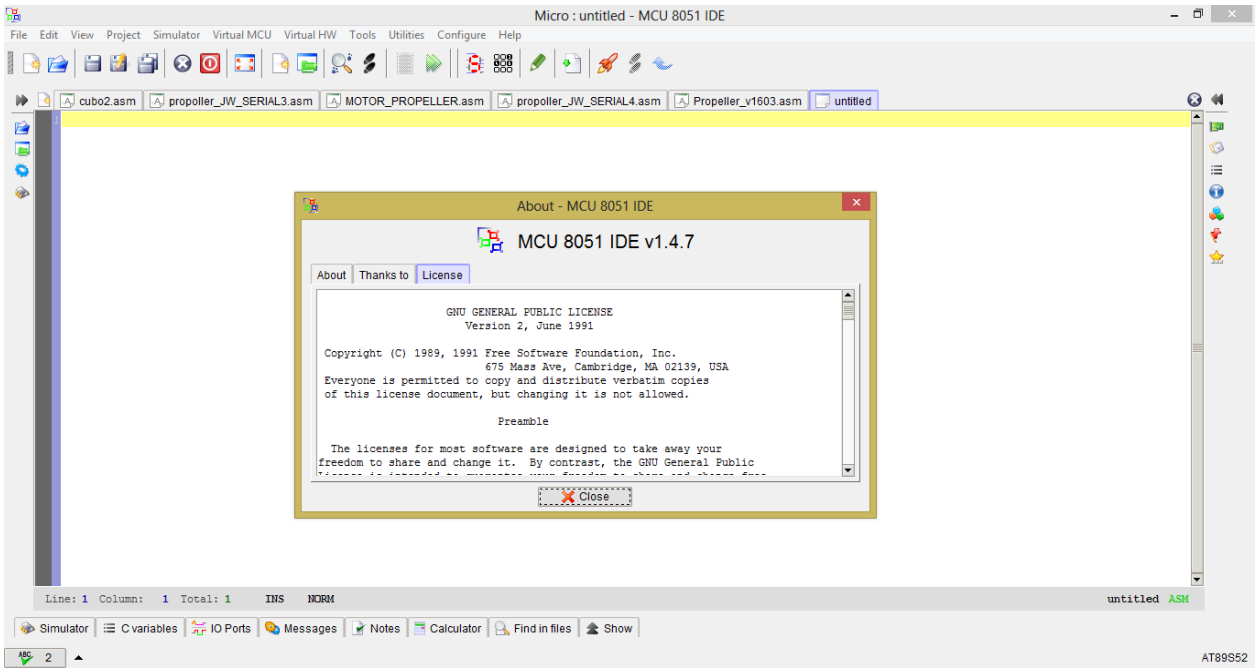


Fig. 1.6 – Tela inicial do simulador MCU 8051

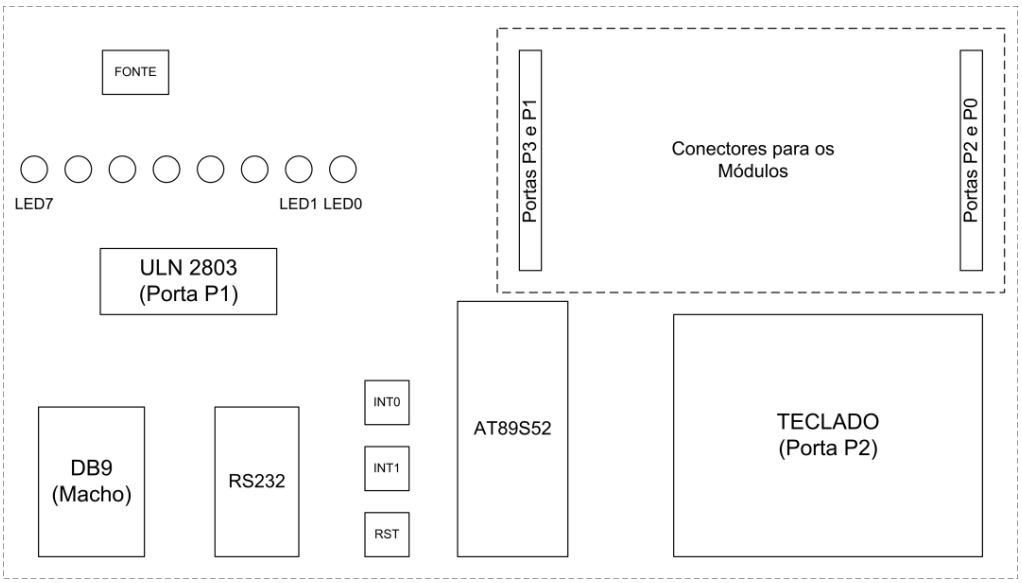


Fig. 1.7 - Esquemático com os componentes do Kit didático

O kit didático necessita de um módulo de gravação em separado. O módulo usado é o ChipMax2. O programa usado para gravação no ChipMax2 é o MaxLoader, cuja tela inicial é mostrada na Fig. 1.8.

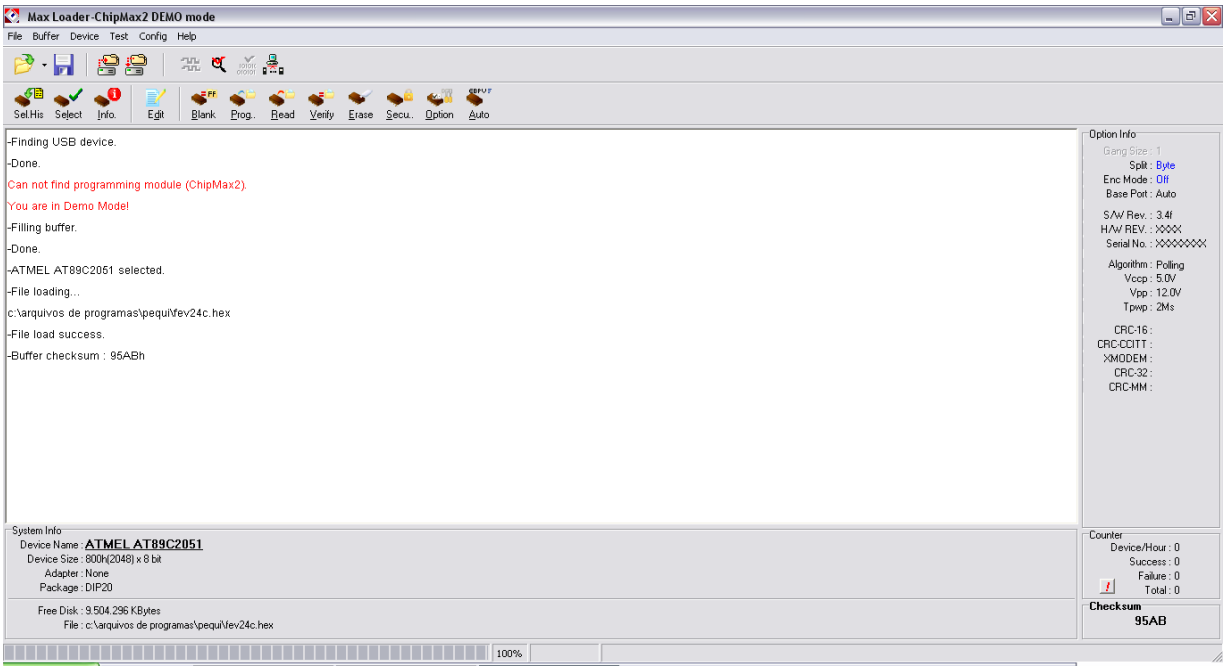


Fig. 1.8 – Tela inicial do programa MaxLoader

1.8 Teclado

Um dos módulos já disponíveis no kit didático é o Teclado de 16 dígitos. As conexões do teclado são mostradas na Fig. 1.9. As teclas são conectadas na forma de uma matriz 4 x 4. As linhas 1 a 4 são conectadas, respectivamente aos pinos P2.7, P2.6, P2.5 e P2.4. As colunas 1 a 4 são conectadas, respectivamente aos pinos P2.3, P2.2, P2.1 e P2.0. Embora seja possível utilizar um driver para o Teclado, tipo 74HC922, utiliza-se aqui o princípio de varredura com o teclado. Nesse procedimento, todos os pinos da porta P2 estão, inicialmente, em nível lógico alto. Assim, por exemplo, coloca-se a Linha L1 em nível lógico baixo (L1=0), através da instrução CLR L1 e verifica-se (varredura) o estado das colunas C1 a C4, através das instruções “JB C1, desvioC1”, até “JB C4, desvioC4”.

Verifica-se, incialmente, a coluna C1. Se ela estiver em nível lógico baixo, significa que a tecla "1" foi pressionada e o processamento desvia para a linha seguinte, onde executa-se o que se deseja

quando a tecla “1” for pressionada. Se C1 estiver em nível lógico alto, desvia para “desvioC1”, para verificar as outras colunas, de C2 a C4.

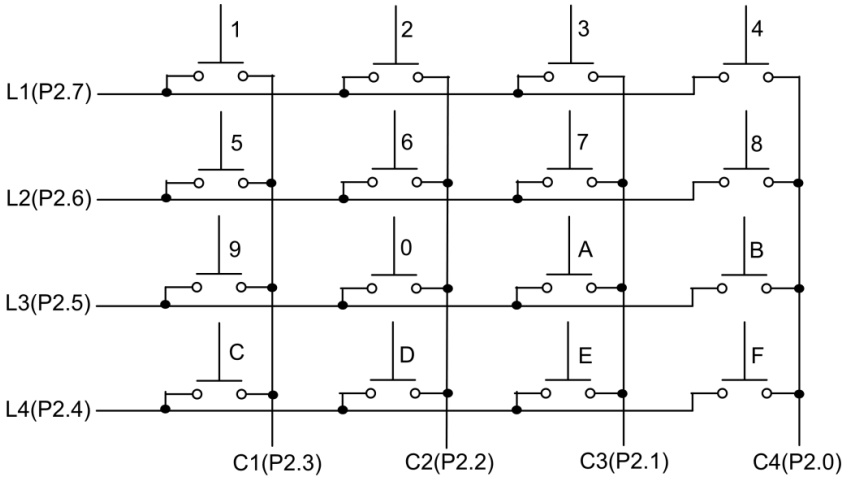


Fig. 1.9 - Teclado do kit didático

Dessa forma, a varredura completa do teclado é feita colocando-se cada linha em nível lógico baixo, uma a uma, as linhas L1 a L4 e verificando, uma a uma, o estado das colunas C1 a C4.

1.9 Conjunto de LEDs

A capacidade de corrente do microcontrolador 8051 de 40 pinos não é suficiente para acionar diretamente um LED, cuja corrente prevista está em torno de 10 mA. Assim, é utilizado o drive ULN2803, mostrado na Fig. 1.10.

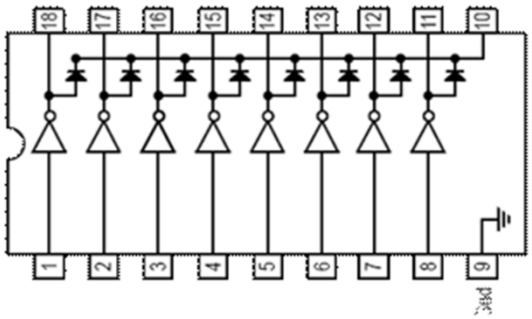


Fig. 1.10 - drive ULN2803 para o motor de passo

Os 8 LEDs são, portanto, conectados à porta P1 através do drive ULN2803.

2 Tarefas do Experimento 1

2.1 Tarefa 1 – Uso de um Simulador Digital

Passo 1: Inicialize o simulador **MCU 8051** e digite o programa mostrado na Tabela 2.1 Pode ser com letras maiúsculas ou minúsculas. Não há necessidade de digitar os comentários após o sinal de ";".

Tabela 2.1 - Onda quadrada no pino P1.0

Rótulo	Mnemônico	Comentários
	ORG 00H	; Diretiva que "diz" ao compilador o endereço da próxima instrução
	LJMP ONDA	; Pula para o endereço " ONDA "
	ORG 30H	; Diretiva que "diz" ao compilador o endereço da próxima instrução
ONDA:	CPL P1.0	; Complementa o pino 0 da porta P1
	MOV R0,#50	; Carrega o registrador R0 com o valor decimal 50
	DJNZ R0,$	; Decrementa o registrador R0 até ele zerar. O desvio é para a própria instrução
	SJMP ONDA	; Desvia para o endereço " ONDA "
	END	; Encerra o programa

Passo 2: Pressione o botão de "**Salvar**" e escolha um nome para o programa (automaticamente o programa terá extensão ".asm"). Em seguida pressione o ícone de "**Compilar**".

Passo 3: Veja o procedimento para simular o programa. A Fig. 2.1 mostra um instante da simulação usando o programa MCU 8051. Observe o **pino 0 da porta P1**, em destaque. Ela estará mudando de 0 para 1 e 1 para 0 em uma frequência constante, que é definida pelo atraso de tempo correspondente a R0 = 50. O resultado é uma onda quadrada no pino P1.0.

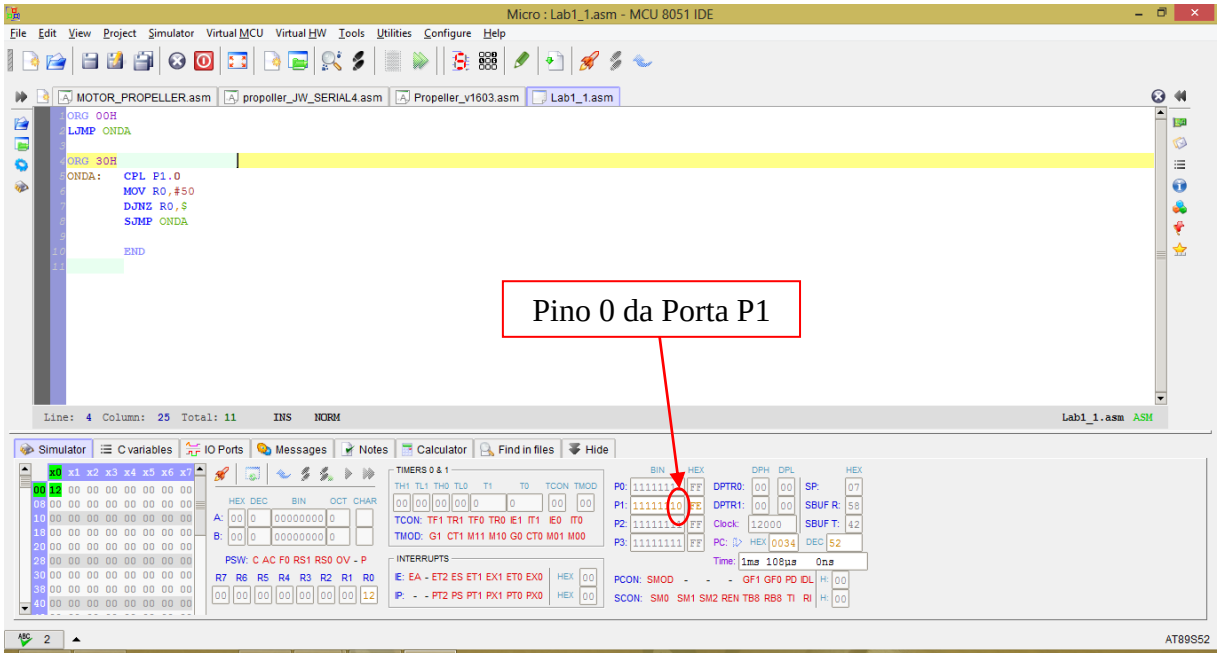


Fig. 2.1 – Visão de um instante durante a simulação com o MCU 8051

Passo 4: Complete a Tabela 2.2 para calcular o período e a frequência da onda quadrada resultante no pino P1.0. A frequência do cristal oscilador é de 11,0592 MHz e, consequentemente, o período do ciclo de máquina ($T = 12/f_{clock}$) é 1,085 µs.

Tabela 2.2 - Cálculo do período da onda quadrada no pino P1.0

Linha	Instrução	Tempo de cada ciclo de máquina	Número de ciclos de máquina da instrução	Qtde de execuções de cada instrução	Tempo total da execução
1	ONDA: CPL P1.0	1,085 µs	1	1	1,085 µs
2	MOV R0,#50	1,085 µs	1		
3	DJNZ R0,$	1,085 µs	2		
4	SJMP ONDA	1,085 µs	2		
	Tempo que corresponde a meio período da onda quadrada (execução da linha 1 à linha 4)				
	Período da onda quadrada (cada duas execuções da linha 1 à linha 4)				
	Frequência da onda quadrada				

Passo 5: Preencha a Tabela 2.3 com os valores calculados:

Tabela 2.3 - Valores calculados para período e frequência

Valor de R0 (decimal)	Período calculado (µs)	Frequência calculada (kHz)
50		

2.2 Tarefa 2 – Uso do Simulador Proteus ou Equivalente

Passo 1: Encerre a simulação no MCU 8051 e inicialize o "Kit Didático Virtual" (Fig. 2.2), desenvolvido no programa "**Proteus**". Esse programa deverá ser usado aqui para visualizar a onda quadrada gerada no pino P1.0. O osciloscópio digital do Proteus deve ser usado para mostrar a onda quadrada no pino P1.0 do microcontrolador. Utilize o canal A do osciloscópio.

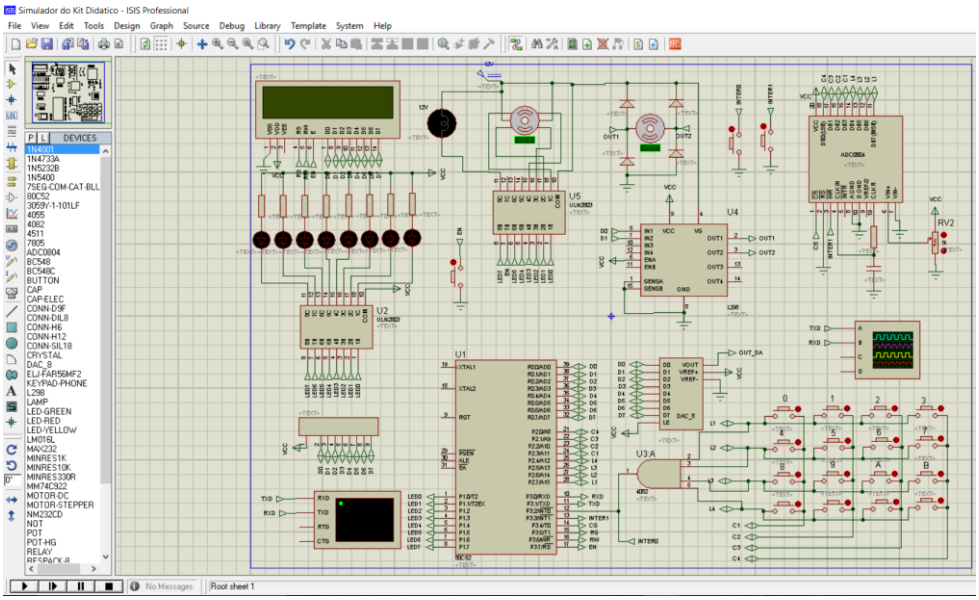


Fig. 2.2 – Tela inicial do Simulador Proteus

Passo 2: "Click" duas vezes em cima do microcontrolador para abrir a janela que permitirá carregar o programa correspondente à onda quadrada. Defina a frequência de clock para 11.0592 MHz e carregue o programa, no formato ".hex", através da janela "Program File", em destaque na Fig. 2.3.

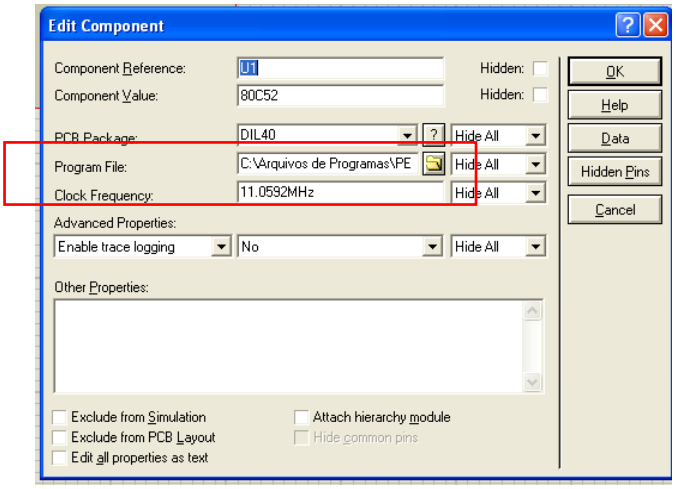


Fig. 2.3 – Janela onde a frequência é definida e o programa ".hex" é carregado

Passo 3: Colocar o programa em funcionamento e observar a onda quadrada no osciloscópio (Fig. 2.4). Meça o período da onda com o auxílio de cursores e preencha a Tabela 2.4.

Tabela 2.4 - Valores medidos no Proteus

Valor de R0 (decimal)	Período medido (µs)	Frequência do sinal (kHz)
50		

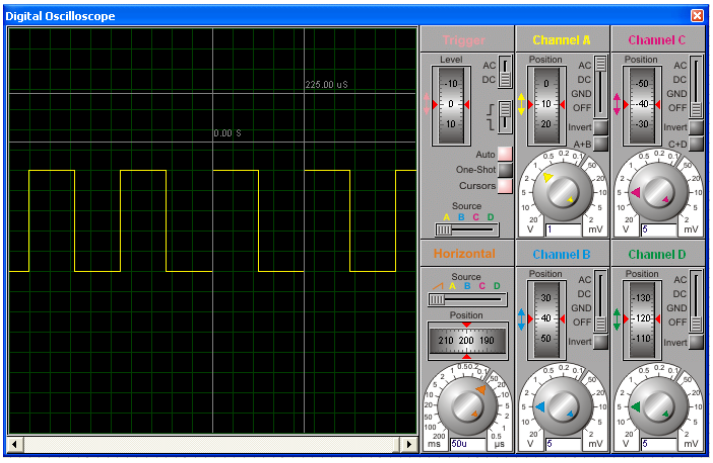


Fig. 2.4 – Onda quadrada no osciloscópio do Proteus

2.3 Tarefa 3 – Uso do Kit Didático

O microcontrolador deverá ser gravado através de uma gravadora separada do Kit. Os passos para gravação e execução do programa no kit são dados a seguir. A gravadora disponível hoje é a **ChipMax2**.

Passo 1: Inicialize o programa "**MaxLoader**", usado para gravação do programa no microcontrolador. Na sequência, apague o programa presente no microcontrolador, carregue o programa ".hex" a ser gravado e use o ícone "prog" para transferir o programa. Alternativamente, após carregar o programa, use o ícone "**auto**".

Passo 2: Recoloque o microcontrolador no kit didático e use o osciloscópio digital para registrar a onda quadrada no pino P1.0.

Passo 3: Anote na Tabela 2.5 o período e a frequência da onda para R0 = 50. Anote também os valores encontrados através de cálculo (seção 2.1) e através do simulador Proteus (seção 2.2).

Obs.: A frequência do cristal oscilador do kit didático é de 11,0592 MHz.

Tabela 2.5 - Valores medidos no kit didático e no Proteus e calculados na seção 2.1 (R0 = 50)

	Valores Calculados	Valores do Simulador Didático (Proteus)	Valores do Kit Real
Período (µs)			
Frequência (kHz)			

2.4 Tarefa 4 – Uso do Teclado por Varredura – Mapeamento

Passo 1: Digite e compile no simulador MCU 8051 o programa da Tabela 2.6. Não é necessário digitar os comentários.

Obs. A instrução **JB bit, endereço** é uma operação com bit que verifica se o "bit" está setado (nível lógico alto); se o "bit" estiver setado, desvia para o "endereço". Caso contrário, desvia para a próxima linha.

Tabela 2.6: Mapeamento do Teclado, conectado à porta P2

Rótulo	**Mnemônico**	**Comentário**	
	L1 EQU P2.7	; Linha L1 equivale ao pino P2.7	
	L2 EQU P2.6	; Linha L2 equivale ao pino P2.6	
	L3 EQU P2.5	; Linha L3 equivale ao pino P2.5	
	L4 EQU P2.4	; Linha L4 equivale ao pino P2.4	
	C1 EQU P2.3	; Coluna C1 equivale ao pino P2.3	
	C2 EQU P2.2	; Coluna C2 equivale ao pino P2.2	
	C3 EQU P2.1	; Coluna C3 equivale ao pino P2.1	
	C4 EQU P2.0	; Coluna C4 equivale ao pino P2.0	
	ORG 00H	; A instrução a seguir está no endereço 00H	
	LJMP INICIO		
	ORG 30H	; A instrução a seguir está no endereço 30H	
INICIO:	MOV SP,#2FH	; O apontador de pilha assume o valor 2FH	
LINHA_L1:	CLR L1	; Faz L1 = 0 (Limpa Linha L1 ≡ Habilita Linha L1)	Parte 1
	SETB L2	; Faz L2 = 1	
	SETB L3	; Faz L3 = 1	
	SETB L4	; Faz L4 = 1	
C11:	JB C1, C21	; Verifica se C1 = 1. Se C1 = 1, desvia para C21	Parte 2
	MOV P1,#30H	; Se C1 = 0, mostra em P1 (Leds) o valor 30H	
C21:	JB C2, C31	; Verifica se C2 = 1. Se C2 = 1, desvia para C31	Parte 2
	MOV P1,#31H	; Se C2 = 0, mostra em P1 (Leds) o valor 31H	
C31:	JB C3, C41	; Verifica se C3 = 1. Se C3 = 1, desvia para C41	Parte 2
	MOV P1,#32H	; Se C3 = 0, mostra em P1 (Leds) o valor 32H	
C41:	JB C4, LINHA_L2	; Verifica se C4 = 1. Se C4 = 1, desvia para LINHA_L2	Parte 2
	MOV P1,#33H	; Se C4 = 0, mostra em P1 (Leds) o valor 33H	
LINHA_L2:	SETB L1	; Faz L1 = 1	Parte 1
	CLR L2	; Faz L2 = 0 (Limpa Linha L2 ≡ Habilita Linha L2)	
	SETB L3	; Faz L3 = 1	

	SETB L4	; Faz L4 = 1	
C12:	JB C1, C22	; Verifica se C1 = 1. Se C1 = 1, desvia para C22	
	MOV P1,#34H	; Se C1 = 0, mostra em P1 (Leds) o valor 34H	
C22:	JB C2, C32	; Verifica se C2 = 1. Se C2 = 1, desvia para C32	
	MOV P1,#35H	; Se C2 = 0, mostra em P1 (Leds) o valor 35H	
C32:	JB C3, C42	; Verifica se C3 = 1. Se C3 = 1, desvia para C42	
	MOV P1,#36H	; Se C3 = 0, mostra em P1 (Leds) o valor 36H	
C42:	JB C4, LINHA_L3	; Verifica se C4 = 1. Se C4 = 1, desvia para LINHA_L3	
	MOV P1,#37H	; Se C4 = 0, mostra em P1 (Leds) o valor 37H	
LINHA_L3:	SETB L1	; Faz L1 = 1	Parte 1
	SETB L2	; Faz L2 = 1	
	CLR L3	; Faz L3 = 0 (Limpa Linha L3 ≡ Habilita Linha L3)	
	SETB L4	; Faz L4 = 1	
C13:	JB C1, C23	; Verifica se C1 = 1. Se C1 = 1, desvia para C23	
	MOV P1,#38H	; Se C1 = 0, mostra em P1 (Leds) o valor 38H	
C23:	JB C2, C33	; Verifica se C2 = 1. Se C2 = 1, desvia para C33	
	MOV P1,#39H	; Se C2 = 0, mostra em P1 (Leds) o valor 39H	
C33:	JB C3, C43	; Verifica se C3 = 1. Se C3 = 1, desvia para C43	
	MOV P1,#41H	; Se C3 = 0, mostra em P1 (Leds) o valor 41H (ASCII da letra A)	
C43:	JB C4, LINHA_L4	; Verifica se C4 = 1. Se C4 = 1, desvia para LINHA_L4	
	MOV P1,#42H	; Se C4 = 0, mostra em P1 (Leds) o valor 42H (ASCII da letra B)	
LINHA_L4:	SETB L1	; Faz L1 = 1	Parte 1
	SETB L2	; Faz L2 = 1	
	SETB L3	; Faz L3 = 1	
	CLR L4	; Faz L4 = 0 (Limpa Linha L4 ≡ Habilita Linha L4)	
C14:	JB C1, C24	; Verifica se C1 = 1. Se C1 = 1, desvia para C24	
	MOV P1,#43H	; Se C1 = 0, mostra em P1 (Leds) o valor 43H (ASCII da letra C)	
C24:	JB C2, C34	; Verifica se C2 = 1. Se C2 = 1, desvia para C34	
	MOV P1,#44H	; Se C2 = 0, mostra em P1 (Leds) o valor 44H (ASCII da letra D)	
C34:	JB C3, C44	; Verifica se C3 = 1. Se C3 = 1, desvia para C44	
	MOV P1,#45H	; Se C3 = 0, mostra em P1 (Leds) o valor 45H (ASCII da letra E)	
C44:	JB C4, V1	; Verifica se C4 = 1. Se C4 = 1, desvia para V1	
	MOV P1,#46H	; Se C4 = 0, mostra em P1 (Leds) o valor 46H (ASCII da letra F)	
V1:	LJMP LINHA_L1	; Volta para a LINHA_L1	
	END		

Passo 2: Use o simulador do Kit Didático e o Kit Didático real para executar o programa da Tabela 2.6 e observar a operação do programa. Pressione uma a uma as teclas do Teclado e verifique o resultado nos LEDs.

Qual a finalidade das Partes 1?

Qual a finalidade das Partes 2?

2.5 Tarefa 5 – Uso do Teclado para Produzir Efeitos sobre os Leds

Passo 1: Digite e compile no simulador MCU 8051 o programa da Tabela 2.7 e Simule no Simulador do Kit Didático.

Obs.: A instrução **CJNE R7,#dado8, endereço** compara o conteúdo do registrador R7 com o dado de 8 bits "dado8"; se eles **NÃO forem iguais**, desvia para "endereço". Se eles forem iguais, desvia para a próxima linha.

Tabela 2.7: Efeitos sobre os LEDs na porta P1, usando o Teclado, conectado à porta P2

Ordem	Rótulo	Mnemônico
1		L1 EQU P2.7
2		C1 EQU P2.3
3		C2 EQU P2.2
4		C3 EQU P2.1
5		
6		ORG 00H
7		LJMP INICIO
8		
9		ORG 30H
10	INICIO:	MOV SP,#2FH
11		MOV A,#01H
12	V0:	LCALL TECLADO
13		
14		CJNE R7,#01H,V1
15		SJMP DIREITA
16		
17	V1:	CJNE R7,#02H,V4
18		SJMP ESQUERDA
19		
20	V4:	CJNE R7,#03H,V0
21		SJMP ALTERNA
22		
23	DIREITA:	MOV P1,A
24		RR A
25		LCALL ATRASO
26		LJMP V0
27		
28		
29		

Ordem	Rótulo	Mnemônico
30	ESQUERDA:	MOV P1,A
31		RL A
32		LCALL ATRASO
33		LJMP V0
34		
35	ALTERNA:	MOV P1,#55H
36		LCALL ATRASO
37		MOV P1,#0AAH
38		LCALL ATRASO
39		LJMP V0
40		
41	TECLADO:	CLR L1
42		JB C1, C12
43		MOV R7,#01H
44	C12:	JB C2, C13
45		MOV R7,#02H
46	C13:	JB C3,SAI
47		MOV R7,#03H
48	SAI:	SETB L1
49		RET
50		
51	ATRASO:	MOV R0,#10
52	V3:	MOV R1,#100
53	V2:	MOV R2,#200
54		DJNZ R2,$
55		DJNZ R1,V2
56		DJNZ R0,V3
57		RET
58		END

Passo 2: Grave o programa da Tabela 2.7 no microcontrolador do kit didático e verifique o funcionamento do programa.

O que ocorre quando a tecla 1 é pressionada?

O que ocorre quando a tecla 2 é pressionada?

O que ocorre quando a tecla 3 é pressionada?

2.6 Tarefa 6 – Revisão

Explique a função de cada um dos comandos dados na Tabela 2.8.

Tabela 2.8: Função dos comandos utilizados ao longo do experimento

Comandos/diretivas	Função
MOV R0,#10	
MOV A,#0FH	
MOV P1,A	
CPL P1.0	
LCALL ATRASO	
DJNZ R1,$	
DJNZ R0,V1	
RET	
SJMP V2	
LJMP INICIO	
ORG 30H	
CJNE R7,#02H,V4	
JB C1, C24	
RR A	
RL A	
SETB L1	
CLR L1	