**Relatório - Projeto Intermediário**

**Jonathan Sutton e Ricardo Israel**

**1 - Arquitetura Escolhida**

Para a entrega do projeto intermediário, foi optado por utilizar a arquitetura do tipo “acumulador”. Uma vantagem desse tipo de arquitetura é que ela é eficiente em termos de espaço e memória, levando em conta que as operações são feitas utilizando apenas 1 registrador (o acumulador).

**2 - Total de instruções e seus pontos de controle:**

A tabela abaixo resume o total de instruções e sua sintaxe. O projeto foi desenvolvido com as instruções já passadas em sala, ou seja, nenhuma instrução nova foi criada.

OBS: Passa = 10, Soma = 01, Sub = 00

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Instrução** | **Mmnemônico** | **Hab Escrita Retorno** | **JMP** | **RET** | **JSR** | **JEQ** | **Sel Mux** | **Hab\_A** | **Operação** | **habFlag=** | **RD** | **WR** |
| Sem operação | NOP | 0 | 0 | 0 | 0 | 0 | X | 0 | XX | 0 | 0 | 0 |
| Carrega valor da memória para A | LDA | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 | 0 | 1 | 0 |
| Soma A e B e armazena em A | SOMA | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 01 | 0 | 1 | 0 |
| Subtrai A e B e armazena em A | SUB | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 00 | 0 | 1 | 0 |
| Carrega valor imediato para A | LDI | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 10 | 0 | 0 | 0 |
| Salva valor do A para a memória | STA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | XX | 0 | 0 | 1 |
| Desvio de execução | JMP | 0 | 1 | 0 | 0 | 0 | X | 0 | XX | 0 | 0 | 0 |
| Desvio condicional de execução | JEQ | 0 | 0 | 0 | 0 | 1 | X | 0 | XX | 0 | 0 | 0 |
| Comparação | CEQ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00 | 1 | 1 | 0 |
| Chamada de Sub Rotina | JSR | 1 | 0 | 0 | 1 | 0 | 0 | 0 | XX | 0 | 0 | 0 |
| Retorno de Sub Rotina | RET | 0 | 0 | 1 | 0 | 0 | 0 | 0 | XX | 0 | 0 | 0 |

**Tabela 1:** Instruções e pontos de controle utilizados no projeto

**3 – Formato das instruções**

Cada instrução possuirá o seguinte formato:

* 13 bits no total
* 4 de *opcode*
* 9 de endereçamento/imediato

O *opcode* é como se fosse um “código” de 4 bits para o decodificador entender qual instrução ele deve executar. Cada uma das 12 instruções citadas na *Tabela 1* possuem um opcode, são eles:

|  |  |
| --- | --- |
| **Instrução** | **Opcode** |
| NOP | 0000 |
| LDA | 0001 |
| SOMA | 0010 |
| SUB | 0011 |
| LDI | 0100 |
| STA | 0101 |
| JMP | 0110 |
| JEQ | 0111 |
| CEQ | 1000 |
| JSR | 1001 |
| RET | 1010 |

**Tabela 2:** Instruções e seus respectivos opcodes.

Já para o endereçamento temos 9 bits que representam o binário do endereço desejado, que pode variar entre 0 e 511.

Caso seja desejado mandar um valor imediato, o 9º bit é obrigatoriamente 0, ou seja, os valores oscilam entre 0 e 255.

**4 – Fluxo de dados do processador**

**Diagrama, Esquemático

Descrição gerada automaticamente**

**Imagem 1:** Circuito que representa o fluxo de dados do processador

A *Imagem 1* ilustra como é o fluxo de dados da arquitetura “acumulador”, onde o “Reg A” (no qual sua saída entra na entrada A da ULA) é o acumulador, ou seja, carrega valores que serão escritos ou salvos na memória.

**5 – Pontos de controle e suas utilizações**

Cada ponto de controle representa uma instrução para o processador fazer. Abaixo uma descrição mais detalhada de cada uma:

* NOP: Sem operação
* LDA: Carrega o valor presente na posição da memória desejada e salva no acumulador
* SOMA: Soma o valor presente na posição da memória desejada e salva no acumulador. Pode somar também um valor imediato
* SUB: Subtrai o valor presente na posição da memória desejada e salva no acumulador. Pode subtrair também um valor imediato
* LDI: Carrega um valor imediato e salva no acumulador
* STA: Salva o valor do acumulador na posição da memória desejada.
* JMP: Desvio de execução, ou seja, pula para a linha desejada
* JEQ: Desvio de execução condicional, ou seja, pula para a linha desejada se o valor do acumulador for igual o valor da posição de memória desejada.
* CEQ: Comparação, ou seja, compara o valor do acumulador com o valor da posição de memória desejada. Serve como uma espécie de “resposta” para o JEQ.
* JSR: Desvio para subrotina, ou seja, pula para a linha desejada e executa até ser requisitado o retorno.
* RET: Retorno da subrotina, ou seja, volta para 1 linha depois que foi executado o JSR.

**6 – Rascunho do diagrama de conexão do processador com os periféricos**

**Diagrama, Esquemático

Descrição gerada automaticamente**

**Imagem 2:** Diagrama de conexão do processador com os periféricos (chaves, botões, LEDRs e Displays)

Como é ilustrado pela *Imagem 2*, a conexão do processador com os periféricos se dá por dois decodificadores que recebem 3 bits e retornam 8. Sendo assim, dependendo da saída do decodificador 2 (que representa o endereço), haverá uma porta *AND* que será ativada, acessando o periférico desejado. As chaves e os botão estão conectados no barramento de leitura, ou seja, estes escrevem no bit menos significativo desse barramento. Já os displays e os LEDs estão no barramento de escrita.

**7 – Rascunho do mapa de memória**

Cada posição ou faixa de posições de memória têm seu destino reservado. A única mudança em reação às aulas foi a alocação do endereço 509 para o “Limpa Leitura RESET”. A tabela abaixo resume onde acessar cada periférico:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Endereço em decimal** | **Periférico** | **Largura dos Dados** | **Tipo de Acesso** | **Bloco (Página) de memória** |
| 0 – 63 | RAM | 8 bits | Leitura/Escrita | 0 |
| 64 – 127 | Reservado | - | - | 1 |
| 128 – 191 | Reservado | - | - | 2 |
| 192 – 255 | Reservado | - | - | 3 |
| 256 | LEDR0 – LEDR7 | 8 bits | Escrita | 4 |
| 257 | LEDR8 | 1 bit | Escrita | 4 |
| 258 | LEDR9 | 1 bit | Escrita | 4 |
| 259 – 287 | Reservado | - | - | 4 |
| 288 | HEX0 | 4 bits | Escrita | 4 |
| 289 | HEX1 | 4 bits | Escrita | 4 |
| 290 | HEX2 | 4 bits | Escrita | 4 |
| 291 | HEX3 | 4 bits | Escrita | 4 |
| 292 | HEX4 | 4 bits | Escrita | 4 |
| 293 | HEX5 | 4 bits | Escrita | 4 |
| 294 – 319 | Reservado | - | - | 4 |
| 320 | SW0 – SW7 | 8 bits | Leitura | 5 |
| 321 | SW8 | 1 bit | Leitura | 5 |
| 322 | SW9 | 1 bit | Leitura | 5 |
| 323 – 351 | Reservado | - | - | 5 |
| 352 | KEY0 | 1 bit | Leitura | 5 |
| 353 | KEY1 | 1 bit | Leitura | 5 |
| 354 | KEY2 | 1 bit | Leitura | 5 |
| 355 | KEY3 | 1 bit | Leitura | 5 |
| 356 | FPGA\_RESET | 1 bit | Leitura | 5 |
| 357 – 383 | Reservado | - | - | 5 |
| 384 - 447 | Reservado | - | - | 6 |
| 448 - 508 | Reservado | - | - | 7 |
| 509 | Limpa Leitura RESET | - | Escrita | 7 |
| 510 | Limpa Leitura KEY1 | - | Escrita | 7 |
| 511 | Limpa Leitura KEY0 | - | Escrita | 7 |

**Tabela 3:** Mapa de memória do processador

**8 – Fonte do programa em Assembly**

Para elaborar a lógica do contador, foram utilizadas 220 linhas de Assembly. Aqui estão as linhas:

**Reset: #Label para resetar o programa**

**STA @510 #Limpa leitura KEY1**

**STA @511 #Limpa leitura KEY0**

**STA @509 #Limpa leitura Reset**

**LDI $1 #Carrega 1 no acumulador**

**STA @63 #Salva 1 no endereço 63 da RAM**

**LDI $9 #Carrega 9 no acumulador**

**STA @19 #Salva 9 no endereço 19 da RAM**

**LDI $10 #Carrega 10 no acumulador**

**STA @20 #Salva 10 no endereço 20 da RAM**

**LDI $11 #Carrega 11 no acumulador**

**STA @21 #Salva 11 no endereço 21 da RAM**

**LDI $12 #Carrega 12 no acumulador**

**STA @22 #Salva 12 no endereço 22 da RAM**

**LDI $13 #Carrega 13 no acumulador**

**STA @23 #Salva 13 no endereço 23 da RAM**

**LDI $14 #Carrega 14 no acumulador**

**STA @24 #Salva 14 no endereço 24 da RAM**

**LDI $15 #Carrega 15 no acumulador**

**STA @25 #Salva 15 no endereço 25 da RAM**

**LDI $0 #Carrega 0 no acumulador**

**STA @62 #Salva 0 no endereço 62 da RAM**

**STA @258 #Apaga o LED 9**

**STA @257 #Apaga o LED 8**

**STA @256 #Apaga o LED 7 a 0**

**STA @288 #Zera o HEX 0**

**STA @289 #Zera o HEX 1**

**STA @290 #Zera o HEX 2**

**STA @291 #Zera o HEX 3**

**STA @292 #Zera o HEX 4**

**STA @293 #Zera o HEX 5**

**STA @0 #Zera o endereço 0 da RAM (Limite das unidades)**

**STA @1 #Zera o endereço 1 da RAM (Limite das dezenas)**

**STA @2 #Zera o endereço 2 da RAM (Limite das centenas)**

**STA @3 #Zera o endereço 3 da RAM (Limite dos milhares)**

**STA @4 #Zera o endereço 4 da RAM (Limite das dezenas de milhares)**

**STA @5 #Zera o endereço 5 da RAM (Limite das centenas de milhares)**

**STA @10 #Zera o endereço 10 da RAM (Valor atual das unidades)**

**STA @11 #Zera o endereço 11 da RAM (Valor atual das dezenas)**

**STA @12 #Zera o endereço 12 da RAM (Valor atual das centenas)**

**STA @13 #Zera o endereço 13 da RAM (Valor atual dos milhares)**

**STA @14 #Zera o endereço 14 da RAM (Valor atual das dezenas de milhares)**

**STA @15 #Zera o endereço 15 da RAM (Valor atual das centenas de milhares)**

**Start: #Label laco principal**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @63 #Compara o valor do KEY1 com 63 (1)**

**JEQ @SetLim #Se o valor do KEY1 for 1, vai para o label SetLim (Setar Limite)**

**LDA @352 #Carrega o valor do KEY0 no acumulador**

**CEQ @63 #Compara o valor do KEY0 com 63 (1)**

**JEQ @Incremento #Se o valor do KEY0 for 1, vai para o label Incremento (Incrementar)**

**LDA @356 #Carrega o valor do Reset no acumulador**

**CEQ @63 #Compara o valor do Reset com 63 (1)**

**JEQ @Reset #Se o valor do Reset for 1, vai para o label Reset (Reseta o programa)**

**JMP @Start #Se nenhum dos botoes forem clicados, vai para o label Start (Laco principal)**

**SetLim: #Label Setar Limite**

**STA @510 #Limpa leitura KEY1**

**LDA @0 #Carrega o valor do endereço 0 da RAM no acumulador (Limite das unidades)**

**STA @288 #Mostra o valor do acumulador HEX 0 (unidades)**

**LDA @1 #Carrega o valor do endereço 1 da RAM no acumulador (Limite das dezenas)**

**STA @289 #Mostra o valor do acumulador HEX 1 (dezenas)**

**LDA @2 #Carrega o valor do endereço 2 da RAM no acumulador (Limite das centenas)**

**STA @290 #Mostra o valor do acumulador HEX 2 (centenas)**

**LDA @3 #Carrega o valor do endereço 3 da RAM no acumulador (Limite dos milhares)**

**STA @291 #Mostra o valor do acumulador HEX 3 (milhares)**

**LDA @4 #Carrega o valor do endereço 4 da RAM no acumulador (Limite das dezenas de milhares)**

**STA @292 #Mostra o valor do acumulador HEX 4 (dezenas de milhares)**

**LDA @5 #Carrega o valor do endereço 5 da RAM no acumulador (Limite das centenas de milhares)**

**STA @293 #Mostra o valor do acumulador HEX 5 (centenas de milhares)**

**LimUni: #Label Limite das unidades**

**LDA @320 #Carrega valor das chaves SW0 a SW7 no acumulador**

**CEQ @20 #Compara o valor das chaves SW0 a SW7 com 20 (10)**

**JEQ @maxUni #Se o valor das chaves SW0 a SW7 for 10, vai para o label maxUni (Limite maximo das unidades)**

**CEQ @21 #Compara o valor das chaves SW0 a SW7 com 21 (11)**

**JEQ @maxUni #Se o valor das chaves SW0 a SW7 for 11, vai para o label maxUni (Limite maximo das unidades)**

**CEQ @22 #Compara o valor das chaves SW0 a SW7 com 22 (12)**

**JEQ @maxUni #Se o valor das chaves SW0 a SW7 for 12, vai para o label maxUni (Limite maximo das unidades)**

**CEQ @23 #Compara o valor das chaves SW0 a SW7 com 23 (13)**

**JEQ @maxUni #Se o valor das chaves SW0 a SW7 for 13, vai para o label maxUni (Limite maximo das unidades)**

**CEQ @24 #Compara o valor das chaves SW0 a SW7 com 24 (14)**

**JEQ @maxUni #Se o valor das chaves SW0 a SW7 for 14, vai para o label maxUni (Limite maximo das unidades)**

**CEQ @25 #Compara o valor das chaves SW0 a SW7 com 25 (15)**

**JEQ @maxUni #Se o valor das chaves SW0 a SW7 for 15, vai para o label maxUni (Limite maximo das unidades)**

**STA @288 #Mostra o valor do acumulador HEX 0 (unidades)**

**STA @0 #Salva o valor do acumulador no endereço 0 da RAM (Limite das unidades)**

**JMP @UNIOK #Vai para o label UNIOK (Unidades menor que o limite)**

**maxUni: #Label Limite maximo das unidades**

**LDA @19 #Carrega o endereco 19 da RAM no acumulador (9)**

**STA @288 #Mostra o valor do acumulador HEX 0 (unidades)**

**STA @0 #Salva 9 no endereço 0 da RAM (Limite das unidades)**

**UNIOK: #Label Unidades menor que o limite**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @62 #Compara o valor do KEY1 com 62 (0)**

**JEQ @LimUni #Se o valor do KEY1 for 0 (nao setou o limite das unidades), vai para o label LimUni (Limite das unidades)**

**STA @510 #Limpa leitura KEY1**

**LimDez: #Label Limite das dezenas**

**LDA @320 #Carrega valor das chaves SW0 a SW7 no acumulador**

**CEQ @20 #Compara o valor das chaves SW0 a SW7 com 20 (10)**

**JEQ @maxDez #Se o valor das chaves SW0 a SW7 for 10, vai para o label maxDez (Limite maximo das dezenas)**

**CEQ @21 #Compara o valor das chaves SW0 a SW7 com 21 (11)**

**JEQ @maxDez #Se o valor das chaves SW0 a SW7 for 11, vai para o label maxDez (Limite maximo das dezenas)**

**CEQ @22 #Compara o valor das chaves SW0 a SW7 com 22 (12)**

**JEQ @maxDez #Se o valor das chaves SW0 a SW7 for 12, vai para o label maxDez (Limite maximo das dezenas)**

**CEQ @23 #Compara o valor das chaves SW0 a SW7 com 23 (13)**

**JEQ @maxDez #Se o valor das chaves SW0 a SW7 for 13, vai para o label maxDez (Limite maximo das dezenas)**

**CEQ @24 #Compara o valor das chaves SW0 a SW7 com 24 (14)**

**JEQ @maxDez #Se o valor das chaves SW0 a SW7 for 14, vai para o label maxDez (Limite maximo das dezenas)**

**CEQ @25 #Compara o valor das chaves SW0 a SW7 com 25 (15)**

**JEQ @maxDez #Se o valor das chaves SW0 a SW7 for 15, vai para o label maxDez (Limite maximo das dezenas)**

**STA @289 #Mostra o valor do acumulador HEX 1 (dezenas)**

**STA @1 #Salva o valor do acumulador no endereço 1 da RAM (Limite das dezenas)**

**JMP @DEZOK #Vai para o label DEZOK (Dezenas menor que o limite)**

**maxDez: #Label Limite maximo das dezenas**

**LDA @19 #Carrega 19 no acumulador**

**STA @289 #Mostra o valor do acumulador HEX 1 (dezenas)**

**STA @1 #Salva 9 no endereço 1 da RAM (Limite das dezenas)**

**DEZOK: #Label Dezenas menor que o limite**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @62 #Compara o valor do KEY1 com 62 (0)**

**JEQ @LimDez #Se o valor do KEY1 for 0 (nao setou o limite das dezenas), vai para o label LimDez (Limite das dezenas)**

**STA @510 #Limpa leitura KEY1**

**LimCen: #Label Limite das centenas**

**LDA @320 #Carrega valor das chaves SW0 a SW7 no acumulador**

**CEQ @20 #Compara o valor das chaves SW0 a SW7 com 20 (10)**

**JEQ @maxCen #Se o valor das chaves SW0 a SW7 for 10, vai para o label maxCen (Limite maximo das centenas)**

**CEQ @21 #Compara o valor das chaves SW0 a SW7 com 21 (11)**

**JEQ @maxCen #Se o valor das chaves SW0 a SW7 for 11, vai para o label maxCen (Limite maximo das centenas)**

**CEQ @22 #Compara o valor das chaves SW0 a SW7 com 22 (12)**

**JEQ @maxCen #Se o valor das chaves SW0 a SW7 for 12, vai para o label maxCen (Limite maximo das centenas)**

**CEQ @23 #Compara o valor das chaves SW0 a SW7 com 23 (13)**

**JEQ @maxCen #Se o valor das chaves SW0 a SW7 for 13, vai para o label maxCen (Limite maximo das centenas)**

**CEQ @24 #Compara o valor das chaves SW0 a SW7 com 24 (14)**

**JEQ @maxCen #Se o valor das chaves SW0 a SW7 for 14, vai para o label maxCen (Limite maximo das centenas)**

**CEQ @25 #Compara o valor das chaves SW0 a SW7 com 25 (15)**

**JEQ @maxCen #Se o valor das chaves SW0 a SW7 for 15, vai para o label maxCen (Limite maximo das centenas)**

**STA @290 #Mostra o valor do acumulador HEX 2 (centenas)**

**STA @2 #Salva o valor do acumulador no endereço 2 da RAM (Limite das centenas)**

**JMP @CENOK #Vai para o label CENOK (Centenas menor que o limite)**

**maxCen: #Label Limite maximo das centenas**

**LDA @19 #Carrega 19 no acumulador**

**STA @290 #Mostra o valor do acumulador HEX 2 (centenas)**

**STA @2 #Salva 9 no endereço 2 da RAM (Limite das centenas)**

**CENOK: #Label Centenas menor que o limite**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @62 #Compara o valor do KEY1 com 62 (0)**

**JEQ @LimCen #Se o valor do KEY1 for 0 (nao setou o limite das centenas), vai para o label LimCen (Limite das centenas)**

**STA @510 #Limpa leitura KEY1**

**LimMil: #Label Limite dos milhares**

**LDA @320 #Carrega valor das chaves SW0 a SW7 no acumulador**

**CEQ @20 #Compara o valor das chaves SW0 a SW7 com 20 (10)**

**JEQ @maxMil #Se o valor das chaves SW0 a SW7 for 10, vai para o label maxMil (Limite maximo dos milhares)**

**CEQ @21 #Compara o valor das chaves SW0 a SW7 com 21 (11)**

**JEQ @maxMil #Se o valor das chaves SW0 a SW7 for 11, vai para o label maxMil (Limite maximo dos milhares)**

**CEQ @22 #Compara o valor das chaves SW0 a SW7 com 22 (12)**

**JEQ @maxMil #Se o valor das chaves SW0 a SW7 for 12, vai para o label maxMil (Limite maximo dos milhares)**

**CEQ @23 #Compara o valor das chaves SW0 a SW7 com 23 (13)**

**JEQ @maxMil #Se o valor das chaves SW0 a SW7 for 13, vai para o label maxMil (Limite maximo dos milhares)**

**CEQ @24 #Compara o valor das chaves SW0 a SW7 com 24 (14)**

**JEQ @maxMil #Se o valor das chaves SW0 a SW7 for 14, vai para o label maxMil (Limite maximo dos milhares)**

**CEQ @25 #Compara o valor das chaves SW0 a SW7 com 25 (15)**

**JEQ @maxMil #Se o valor das chaves SW0 a SW7 for 15, vai para o label maxMil (Limite maximo dos milhares)**

**STA @291 #Mostra o valor do acumulador HEX 3 (milhares)**

**STA @3 #Salva o valor do acumulador no endereço 3 da RAM (Limite dos milhares)**

**JMP @MILOK #Vai para o label MILOK (Milhares menor que o limite)**

**maxMil: #Label Limite maximo dos milhares**

**LDA @19 #Carrega 19 no acumulador**

**STA @291 #Mostra o valor do acumulador HEX 3 (milhares)**

**STA @3 #Salva 9 no endereço 3 da RAM (Limite dos milhares)**

**MILOK: #Label Milhares menor que o limite**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @62 #Compara o valor do KEY1 com 62 (0)**

**JEQ @LimMil #Se o valor do KEY1 for 0 (nao setou o limite dos milhares), vai para o label LimMil (Limite dos milhares)**

**STA @510 #Limpa leitura KEY1**

**LimDezM: #Label Limite das dezenas de milhares**

**LDA @320 #Carrega valor das chaves SW0 a SW7 no acumulador**

**CEQ @20 #Compara o valor das chaves SW0 a SW7 com 20 (10)**

**JEQ @maxDezM #Se o valor das chaves SW0 a SW7 for 10, vai para o label maxDezM (Limite maximo das dezenas de milhares)**

**CEQ @21 #Compara o valor das chaves SW0 a SW7 com 21 (11)**

**JEQ @maxDezM #Se o valor das chaves SW0 a SW7 for 11, vai para o label maxDezM (Limite maximo das dezenas de milhares)**

**CEQ @22 #Compara o valor das chaves SW0 a SW7 com 22 (12)**

**JEQ @maxDezM #Se o valor das chaves SW0 a SW7 for 12, vai para o label maxDezM (Limite maximo das dezenas de milhares)**

**CEQ @23 #Compara o valor das chaves SW0 a SW7 com 23 (13)**

**JEQ @maxDezM #Se o valor das chaves SW0 a SW7 for 13, vai para o label maxDezM (Limite maximo das dezenas de milhares)**

**CEQ @24 #Compara o valor das chaves SW0 a SW7 com 24 (14)**

**JEQ @maxDezM #Se o valor das chaves SW0 a SW7 for 14, vai para o label maxDezM (Limite maximo das dezenas de milhares)**

**CEQ @25 #Compara o valor das chaves SW0 a SW7 com 25 (15)**

**JEQ @maxDezM #Se o valor das chaves SW0 a SW7 for 15, vai para o label maxDezM (Limite maximo das dezenas de milhares)**

**STA @292 #Mostra o valor do acumulador HEX 4 (dezenas de milhares)**

**STA @4 #Salva o valor do acumulador no endereço 4 da RAM (Limite das dezenas de milhares)**

**JMP @DEZMOK #Vai para o label DEZMOK (Dezenas de milhares menor que o limite)**

**maxDezM: #Label Limite maximo das dezenas de milhares**

**LDA @19 #Carrega 19 no acumulador**

**STA @292 #Mostra o valor do acumulador HEX 4 (dezenas de milhares)**

**STA @4 #Salva 9 no endereço 4 da RAM (Limite das dezenas de milhares)**

**DEZMOK: #Label Dezenas de milhares menor que o limite**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @62 #Compara o valor do KEY1 com 62 (0)**

**JEQ @LimDezM #Se o valor do KEY1 for 0 (nao setou o limite das dezenas de milhares), vai para o label LimDezM (Limite das dezenas de milhares)**

**STA @510 #Limpa leitura KEY1**

**LimCenM: #Label Limite das centenas de milhares**

**LDA @320 #Carrega valor das chaves SW0 a SW7 no acumulador**

**CEQ @20 #Compara o valor das chaves SW0 a SW7 com 20 (10)**

**JEQ @maxCenM #Se o valor das chaves SW0 a SW7 for 10, vai para o label maxCenM (Limite maximo das centenas de milhares)**

**CEQ @21 #Compara o valor das chaves SW0 a SW7 com 21 (11)**

**JEQ @maxCenM #Se o valor das chaves SW0 a SW7 for 11, vai para o label maxCenM (Limite maximo das centenas de milhares)**

**CEQ @22 #Compara o valor das chaves SW0 a SW7 com 22 (12)**

**JEQ @maxCenM #Se o valor das chaves SW0 a SW7 for 12, vai para o label maxCenM (Limite maximo das centenas de milhares)**

**CEQ @23 #Compara o valor das chaves SW0 a SW7 com 23 (13)**

**JEQ @maxCenM #Se o valor das chaves SW0 a SW7 for 13, vai para o label maxCenM (Limite maximo das centenas de milhares)**

**CEQ @24 #Compara o valor das chaves SW0 a SW7 com 24 (14)**

**JEQ @maxCenM #Se o valor das chaves SW0 a SW7 for 14, vai para o label maxCenM (Limite maximo das centenas de milhares)**

**CEQ @25 #Compara o valor das chaves SW0 a SW7 com 25 (15)**

**JEQ @maxCenM #Se o valor das chaves SW0 a SW7 for 15, vai para o label maxCenM (Limite maximo das centenas de milhares)**

**STA @293 #Mostra o valor do acumulador HEX 5 (centenas de milhares)**

**STA @5 #Salva o valor do acumulador no endereço 5 da RAM (Limite das centenas de milhares)**

**JMP @CENMOK #Vai para o label CENMOK (Centenas de milhares menor que o limite)**

**maxCenM: #Label Limite maximo das centenas de milhares**

**LDA @19 #Carrega 19 no acumulador**

**STA @293 #Mostra o valor do acumulador HEX 5 (centenas de milhares)**

**STA @5 #Salva 9 no endereço 5 da RAM (Limite das centenas de milhares)**

**CENMOK: #Label Centenas de milhares menor que o limite**

**LDA @353 #Carrega o valor do KEY1 no acumulador**

**CEQ @62 #Compara o valor do KEY1 com 62 (0)**

**JEQ @LimCenM #Se o valor do KEY1 for 0 (nao setou o limite das centenas de milhares), vai para o label LimCenM (Limite das centenas de milhares)**

**STA @510 #Limpa leitura KEY1**

**LDA @10 #Carrega o valor do endereço 10 da RAM no acumulador (valor atual das unidades)**

**STA @288 #Mostra o valor do acumulador HEX 0 (unidades)**

**LDA @11 #Carrega o valor do endereço 11 da RAM no acumulador (valor atual das dezenas)**

**STA @289 #Mostra o valor do acumulador HEX 1 (dezenas)**

**LDA @12 #Carrega o valor do endereço 12 da RAM no acumulador (valor atual das centenas)**

**STA @290 #Mostra o valor do acumulador HEX 2 (centenas)**

**LDA @13 #Carrega o valor do endereço 13 da RAM no acumulador (valor atual dos milhares)**

**STA @291 #Mostra o valor do acumulador HEX 3 (milhares)**

**LDA @14 #Carrega o valor do endereço 14 da RAM no acumulador (valor atual das dezenas de milhares)**

**STA @292 #Mostra o valor do acumulador HEX 4 (dezenas de milhares)**

**LDA @15 #Carrega o valor do endereço 15 da RAM no acumulador (valor atual das centenas de milhares)**

**STA @293 #Mostra o valor do acumulador HEX 5 (centenas de milhares)**

**JMP @Start #Volta para o label Start (inicio do programa)**

**Incremento: #Label Incremento**

**STA @511 #Limpa leitura KEY0**

**LDA @10 #Carrega o valor do endereço 10 da RAM no acumulador (valor atual das unidades)**

**SOMA @63 #Soma o valor do acumulador com 63 (1)**

**CEQ @20 #Compara o valor do acumulador com o endereco 20 da RAM (10)**

**JEQ @IncDez #Se o valor do acumulador for 10, vai para o label IncDez (Incremento das dezenas)**

**STA @10 #Se o valor do acumulador nao for 10, salva o valor do acumulador no endereço 10 da RAM (valor atual das unidades)**

**JMP @Display #Vai para o label Display (Mostra o valor atual)**

**IncDez: #Label Incremento das dezenas**

**LDA @62 #Carrega o valor 62 (0) no acumulador**

**STA @10 #Zera o valor atual das unidades**

**LDA @11 #Carrega o valor do endereço 11 da RAM no acumulador (valor atual das dezenas)**

**SOMA @63 #Soma o valor do acumulador com 63 (1)**

**CEQ @20 #Compara o valor do acumulador com o endereco 20 da RAM (10)**

**JEQ @IncCen #Se o valor do acumulador for 10, vai para o label IncCen (Incremento das centenas)**

**STA @11 #Se o valor do acumulador nao for 10, salva o valor do acumulador no endereço 11 da RAM (valor atual das dezenas)**

**JMP @Display #Vai para o label Display (Mostra o valor atual)**

**IncCen: #Label Incremento das centenas**

**LDA @62 #Carrega o valor 62 (0) no acumulador**

**STA @11 #Zera o valor atual das dezenas**

**LDA @12 #Carrega o valor do endereço 12 da RAM no acumulador (valor atual das centenas)**

**SOMA @63 #Soma o valor do acumulador com 63 (1)**

**CEQ @20 #Compara o valor do acumulador com o endereco 20 da RAM (10)**

**JEQ @IncMil #Se o valor do acumulador for 10, vai para o label IncMil (Incremento dos milhares)**

**STA @12 #Se o valor do acumulador nao for 10, salva o valor do acumulador no endereço 12 da RAM (valor atual das centenas)**

**JMP @Display #Vai para o label Display (Mostra o valor atual)**

**IncMil: #Label Incremento dos milhares**

**LDA @62 #Carrega o valor 62 (0) no acumulador**

**STA @12 #Zera o valor atual das centenas**

**LDA @13 #Carrega o valor do endereço 13 da RAM no acumulador (valor atual dos milhares)**

**SOMA @63 #Soma o valor do acumulador com 63 (1)**

**CEQ @20 #Compara o valor do acumulador com o endereco 20 da RAM (10)**

**JEQ @IncDezM #Se o valor do acumulador for 10, vai para o label IncDezM (Incremento das dezenas de milhares)**

**STA @13 #Se o valor do acumulador nao for 10, salva o valor do acumulador no endereço 13 da RAM (valor atual dos milhares)**

**JMP @Display #Vai para o label Display (Mostra o valor atual)**

**IncDezM: #Label Incremento das dezenas de milhares**

**LDA @62 #Carrega o valor 62 (0) no acumulador**

**STA @13 #Zera o valor atual dos milhares**

**LDA @14 #Carrega o valor do endereço 14 da RAM no acumulador (valor atual das dezenas de milhares)**

**SOMA @63 #Soma o valor do acumulador com 63 (1)**

**CEQ @20 #Compara o valor do acumulador com o endereco 20 da RAM (10)**

**JEQ @IncCenM #Se o valor do acumulador for 10, vai para o label IncCenM (Incremento das centenas de milhares)**

**STA @14 #Se o valor do acumulador nao for 10, salva o valor do acumulador no endereço 14 da RAM (valor atual das dezenas de milhares)**

**JMP @Display #Vai para o label Display (Mostra o valor atual)**

**IncCenM: #Label Incremento das centenas de milhares**

**LDA @62 #Carrega o valor 62 (0) no acumulador**

**STA @14 #Zera o valor atual das dezenas de milhares**

**LDA @15 #Carrega o valor do endereço 15 da RAM no acumulador (valor atual das centenas de milhares)**

**SOMA @63 #Soma o valor do acumulador com 63 (1)**

**CEQ @20 #Compara o valor do acumulador com o endereco 20 da RAM (10)**

**JEQ @NoveNove #Se o valor do acumulador for 10, vai para o label NoveNove (valor nos displays igual a 999999)**

**STA @15 #Se o valor do acumulador nao for 10, salva o valor do acumulador no endereço 15 da RAM (valor atual das centenas de milhares)**

**JMP @Display #Vai para o label Display (Mostra o valor atual)**

**Display: #Label Mostra o valor atual**

**LDA @10 #Carrega o valor do endereço 10 da RAM no acumulador (valor atual das unidades)**

**STA @288 #Salva o valor do acumulador no HEX 0**

**LDA @11 #Carrega o valor do endereço 11 da RAM no acumulador (valor atual das dezenas)**

**STA @289 #Salva o valor do acumulador no HEX 1**

**LDA @12 #Carrega o valor do endereço 12 da RAM no acumulador (valor atual das centenas)**

**STA @290 #Salva o valor do acumulador no HEX 2**

**LDA @13 #Carrega o valor do endereço 13 da RAM no acumulador (valor atual dos milhares)**

**STA @291 #Salva o valor do acumulador no HEX 3**

**LDA @14 #Carrega o valor do endereço 14 da RAM no acumulador (valor atual das dezenas de milhares)**

**STA @292 #Salva o valor do acumulador no HEX 4**

**LDA @15 #Carrega o valor do endereço 15 da RAM no acumulador (valor atual das centenas de milhares)**

**STA @293 #Salva o valor do acumulador no HEX 5**

**JMP @CompLim #Vai para o label CompLim (Compara o valor atual com o valor do limite)**

**NoveNove: #Label valor chegou a 999999**

**LDA @63 #Carrega o valor 63 (1) no acumulador**

**STA @258 #Ascende o LED 9**

**LDA @356 #Carrega o valor do Reset no acumulador**

**CEQ @63 #Compara o valor do acumulador com 63 (1)**

**JEQ @Reset #Se o valor do acumulador for 1, vai para o label Reset (Reseta o programa)**

**JMP @NoveNove #Se o valor do acumulador nao for 1, vai para o label NoveNove**

**CompLim: #Label Compara o valor atual com o valor do limite**

**LDA @10 #Carrega o valor do endereço 10 da RAM no acumulador (valor atual das unidades)**

**CEQ @0 #Compara o valor do acumulador com o endereco 0 da RAM (limite das unidades)**

**JEQ @CompDez #Se for igual, vai para o label CompDez (Compara o valor atual das dezenas com o valor do limite das dezenas)**

**JMP @Start #Se nao for igual, vai para o label Start (Loop principal)**

**CompDez: #Label Compara o valor atual com o valor do limite das dezenas**

**LDA @11 #Carrega o valor do endereço 11 da RAM no acumulador (valor atual das dezenas)**

**CEQ @1 #Compara o valor do acumulador com o endereco 1 da RAM (limite das dezenas)**

**JEQ @CompCen #Se for igual, vai para o label CompCen (Compara o valor atual das centenas com o valor do limite das centenas)**

**JMP @Start #Se nao for igual, vai para o label Start (Loop principal)**

**CompCen: #Label Compara o valor atual com o valor do limite das centenas**

**LDA @12 #Carrega o valor do endereço 12 da RAM no acumulador (valor atual das centenas)**

**CEQ @2 #Compara o valor do acumulador com o endereco 2 da RAM (limite das centenas)**

**JEQ @CompMil #Se for igual, vai para o label CompMil (Compara o valor atual dos milhares com o valor do limite dos milhares)**

**JMP @Start #Se nao for igual, vai para o label Start (Loop principal)**

**CompMil: #Label Compara o valor atual com o valor do limite dos milhares**

**LDA @13 #Carrega o valor do endereço 13 da RAM no acumulador (valor atual dos milhares)**

**CEQ @3 #Compara o valor do acumulador com o endereco 3 da RAM (limite dos milhares)**

**JEQ @CompDezM #Se for igual, vai para o label CompDezM (Compara o valor atual das dezenas de milhares com o valor do limite das dezenas de milhares)**

**JMP @Start #Se nao for igual, vai para o label Start (Loop principal)**

**CompDezM: #Label Compara o valor atual com o valor do limite das dezenas de milhares**

**LDA @14 #Carrega o valor do endereço 14 da RAM no acumulador (valor atual das dezenas de milhares)**

**CEQ @4 #Compara o valor do acumulador com o endereco 4 da RAM (limite das dezenas de milhares)**

**JEQ @CompCenM #Se for igual, vai para o label CompCenM (Compara o valor atual das centenas de milhares com o valor do limite das centenas de milhares)**

**JMP @Start #Se nao for igual, vai para o label Start (Loop principal)**

**CompCenM: #Label Compara o valor atual com o valor do limite das centenas de milhares**

**LDA @15 #Carrega o valor do endereço 15 da RAM no acumulador (valor atual das centenas de milhares)**

**CEQ @5 #Compara o valor do acumulador com o endereco 5 da RAM (limite das centenas de milhares)**

**JEQ @Fim #Se for igual, vai para o label Fim (Fim do programa)**

**JMP @Start #Se nao for igual, vai para o label Start (Loop principal)**

**Fim: #Label Fim do programa**

**LDA @63 #Carrega o valor 63 (1) no acumulador**

**STA @258 #Ascende o LED 9**

**STA @257 #Ascende o LED 8**

**LDI $255 #Carrega o valor 255 no acumulador**

**STA @256 #Ascende os LEDs 7 a 0**

**LDA @356 #Carrega o valor do Reset no acumulador**

**CEQ @63 #Compara o valor do acumulador com 63 (1)**

**JEQ @Reset #Se o valor do acumulador for 1, vai para o label Reset (Reseta o programa)**

**JMP @Fim #Se o valor do acumulador nao for 1, vai para o label Fim**

**9 – Como testar o contador**

Após compilar o programa, já é possível testar o contador na placa.

* Caso queira contar sem definir um limite prévio, é só ir apertando KEY0 para incrementar o número no display
* Caso queira definir um limite prévio, clicar em KEY1 para começar a definir o limite. Após clicar na primeira vez, não acontece nada, mas já é possível mexer nas chaves e ver qual será o limite das unidades, com o limite máximo de 9. Ou seja, caso seja passado um valor maior do que 9 pelas chaves, o limite salvo será 9. A seguir, clicar no botão KEY1 para salvar a escolha e passar a alterar o limite das dezenas, e assim por diante, até chegar nas centenas de milhares.
* Caso queira começar a contar, e no meio da contagem definir um limite prévio, é só apertar KEY1 a qualquer momento da contagem e definir o limite, da mesma maneira descrita anteriormente
* OBS: o limite precisa ser maior do que o número atual da contagem, e precisará ser setado as 6 casas decimais obrigatoriamente, antes de mais nada.
* É possível também resetar o programa a qualquer momento, clicando no botão FPGA\_RESET (exceto se estiver no meio da setagem do limite).
* Também será possível MUDAR o limite no meio da contagem.