UNIVERSIDADE FEDERAL DE VIÇOSA

CIÊNCIA DA COMPUTAÇÃO

**INTRODUÇÃO AOS SISTEMAS LÓGICOS DIGITAIS**

# TRABALHO PRÁTICO CODIFICADOR MORSE

FLORESTAL

2018

Este trabalho prático consiste na implementação de um codificador Morse capaz de converter um número (de 0 a 9) em seu sinal correspondente no código Morse.

O trabalho foi realizado em grupo pelos seguintes alunos:

* **Yuri Dimitre - 3485**
* **Samuel Pedro - 3494**
* **William Lucas - 3472**
* **Marcos Túlio – 3504**

Todo o desenvolvimento dos módulos e esquemas no *Logisim* se encontram disponibilizados no GitHub: <https://github.com/Numb4r/codigo-morse-verilog> e se encontram licenciados pela licença GPL v3.0.

**Tabela da Verdade**

Inicialmente no desenvolvimento do módulo foi realizado o levantamento das equações booleanas para cada saída através da tabela da verdade utilizando mapas de Karnaugh.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | S1 | S2 | S3 | S4 | S5 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | x | x | x | x | x |
| 1 | 0 | 1 | 1 | x | x | x | x | x |
| 1 | 1 | 0 | 0 | x | x | x | x | x |
| 1 | 1 | 0 | 1 | x | x | x | x | x |
| 1 | 1 | 1 | 0 | x | x | x | x | x |
| 1 | 1 | 1 | 1 | x | x | x | x | x |

**Ps: Traço = 0**

**Ponto = 1.**

1. **Mapas de cada saída:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S1: | C’.D’ | C’.D | C.D | C.D’ |
| A’.B’ | 0 | 1 | 1 | 1 |
| A’.B | 1 | 1 | 0 | 0 |
| A.B | x | x | x | x |
| A.B’ | 0 | 0 | x | x |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S2: | C’.D’ | C’.D | C.D | C.D’ |
| A’.B’ | 0 | 0 | 1 | 1 |
| A’.B | 1 | 1 | 0 | 1 |
| A.B | x | x | x | x |
| A.B’ | 0 | 0 | x | x |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S3: | C’.D’ | C’.D | C.D | C.D’ |
| A’.B’ | 0 | 0 | 1 | 0 |
| A’.B | 1 | 1 | 1 | 1 |
| A.B | x | x | x | x |
| A.B’ | 0 | 0 | x | x |

**S1 = B’.C + B.C’ + A’.B’.D S2 = B’.C + C.D’ + B.C’ S3 = B + C.D**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S4: | C’.D’ | C’.D | C.D | C.D’ |
| A’.B’ | 0 | 0 | 0 | 0 |
| A’.B | 1 | 1 | 1 | 1 |
| A.B | x | x | x | x |
| A.B’ | 1 | 0 | x | x |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S5: | C’.D’ | C’.D | C.D | C.D’ |
| A’.B’ | 0 | 0 | 0 | 0 |
| A’.B | 0 | 1 | 1 | 1 |
| A.B | x | x | x | x |
| A.B’ | 1 | 1 | x | x |

**S4 = B + A.D’ S5 = A + B.D + B.C**

1. **Formas Canônicas:**
   1. **Soma de Produtos:**

S1 (A,B,C,D) = ∑m(1, 2, 3, 4, 5)

S2 (A,B,C,D) = ∑m(2, 3, 4, 5, 6)

S3 (A,B,C,D) = ∑m(3, 4, 5, 6, 7)

S4 (A,B,C,D) = ∑m(4, 5, 6, 7, 8)

S5 (A,B,C,D) = ∑m(5, 6, 7, 8, 9)

* 1. **Produto das Somas:**

S1 (A,B,C,D) = πm(0, 6, 7, 8, 9)

S2 (A,B,C,D) = πm (0, 1, 7, 8, 9)

S3 (A,B,C,D) = πm (0, 1, 2, 8, 9)

S4 (A,B,C,D) = πm (0, 1, 2, 3, 9)

S5 (A,B,C,D) = πm (0, 1, 2, 3, 4)

1. **Mintermos:**

**S1 =** A’B’C’D + A’B’CD’ + A’B’CD + A’BC’D’+ A’BC’D

**S2 =** A’B’CD’ + A’B’CD + A’BC’D’+ A’BC’D + A’BCD’

**S3 =** A’B’CD + A’BC’D’+ A’BC’D + A’BCD’ + A’BCD

**S4 =** A’BC’D’+ A’BC’D + A’BCD’ + A’BCD + AB’C’D’

**S5 =** A’BC’D + A’BCD’ + A’BCD + AB’C’D’ + AB’C’D

1. **Maxtermos**

**S1 =** (A + B + C + D) (A + B’ + C’ + D) (A + B’ + C’ + D’) (A’ + B + C + D) (A’ + B + C + D’)

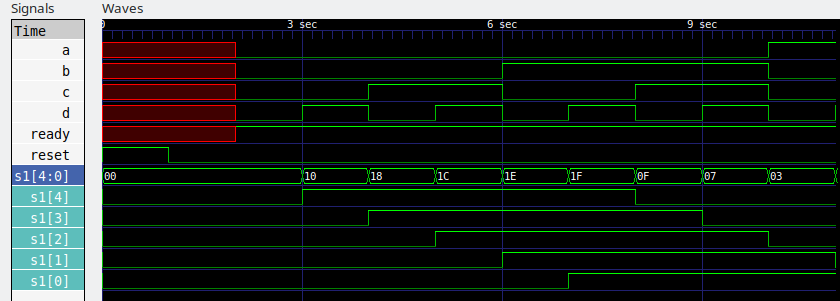
**S2 =** (A + B + C + D) (A + B + C + D’) (A + B’ + C’ + D’) (A’ + B + C + D) (A’ + B + C + D’)

**S3 =** (A + B + C + D) (A + B + C + D’) (A + B + C’ + D) (A’ + B + C + D) (A + B + C + D’)

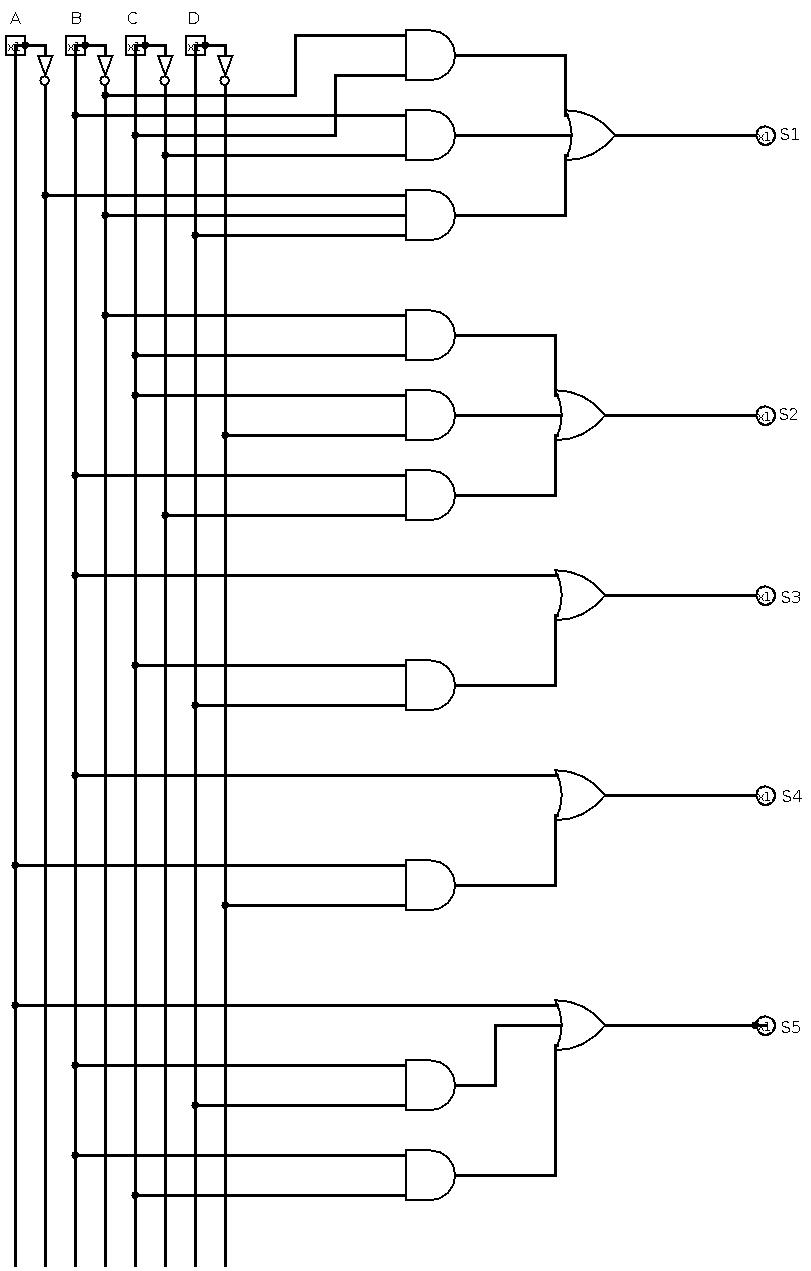
**S4 =** (A + B + C + D) (A + B + C + D’) (A + B + C’ + D) (A + B + C’ + D’) (A’ + B + C + D’)

**S5 =** (A + B + C + D) (A + B + C + D’) (A + B + C’ + D) (A + B + C’ + D’) (A + B’ + C + D)

1. **Formas de onda**



1. **Circuito simplificado com portas lógicas no *LOGISIM***

****

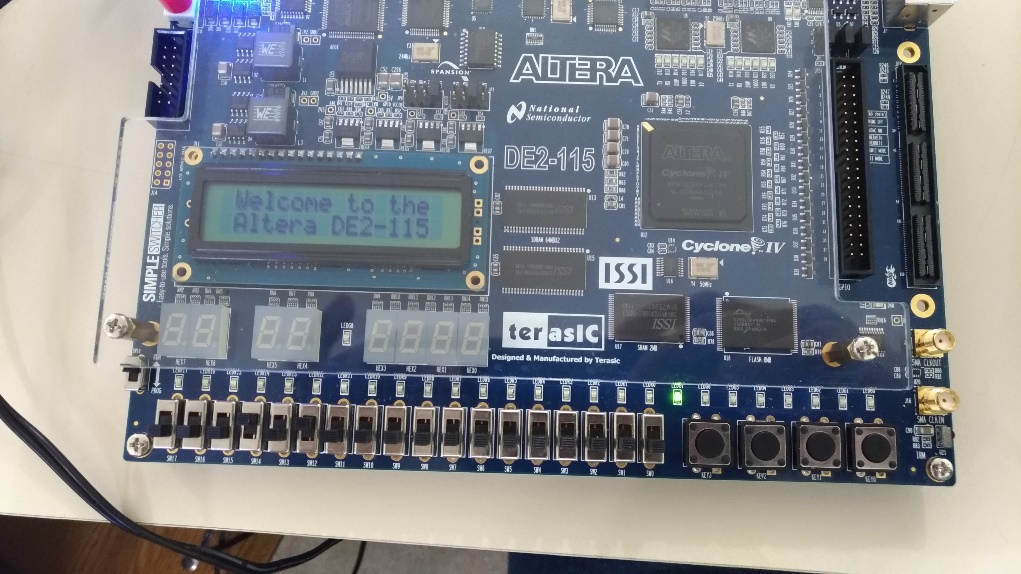
1. **Implementação em FPGA**

A implementação em FPGA foi realizada com sucesso com o auxílio do monitor da disciplina em um modelo de FPGA Altera DE2-115 e seguiu os seguintes critérios:

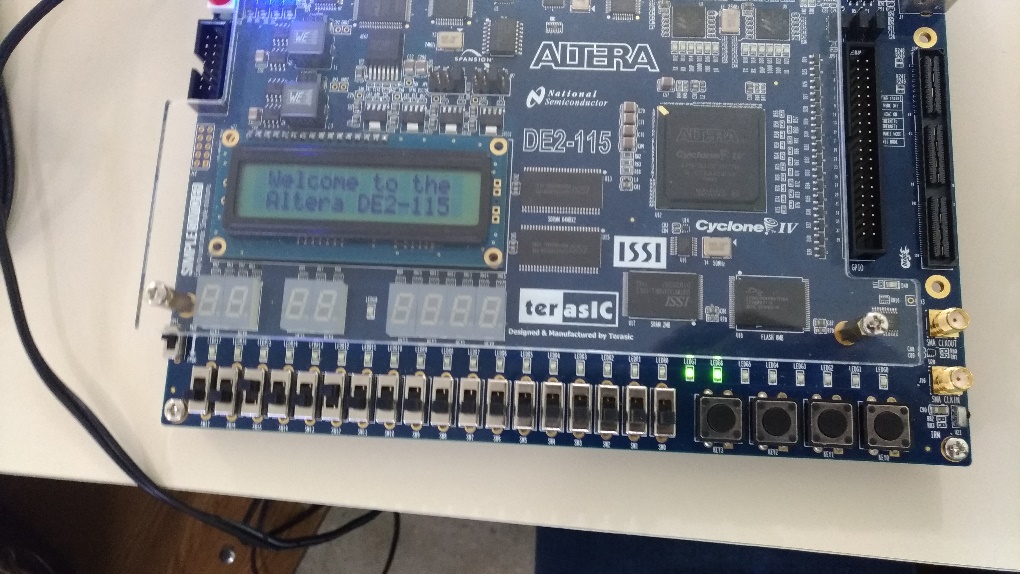
* A codificação do código de binário para Morse apenas iniciará quando o switch correspondente ao *ready* estiver ativo.
* Em caso da entrada em binário for superior ou igual ao número 1010(10 em decimal), a entrada será considerada como invalida para a conversão e assim nenhum led ficará ativo.
* Caso o switch correspondente ao *reset* seja ativado, todas as saídas serão zeradas e permanecerão assim até que o mesmo seja desativado.

As seguintes imagens ilustram o funcionamento exato do modulo de FPGA com o codificador implementado:

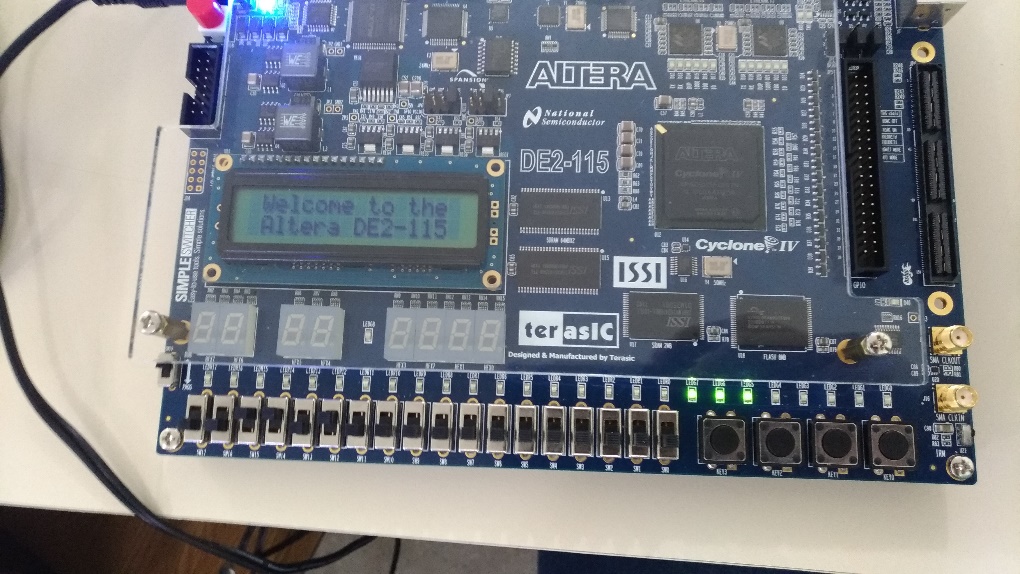
* Exemplo da conversão do número 1:



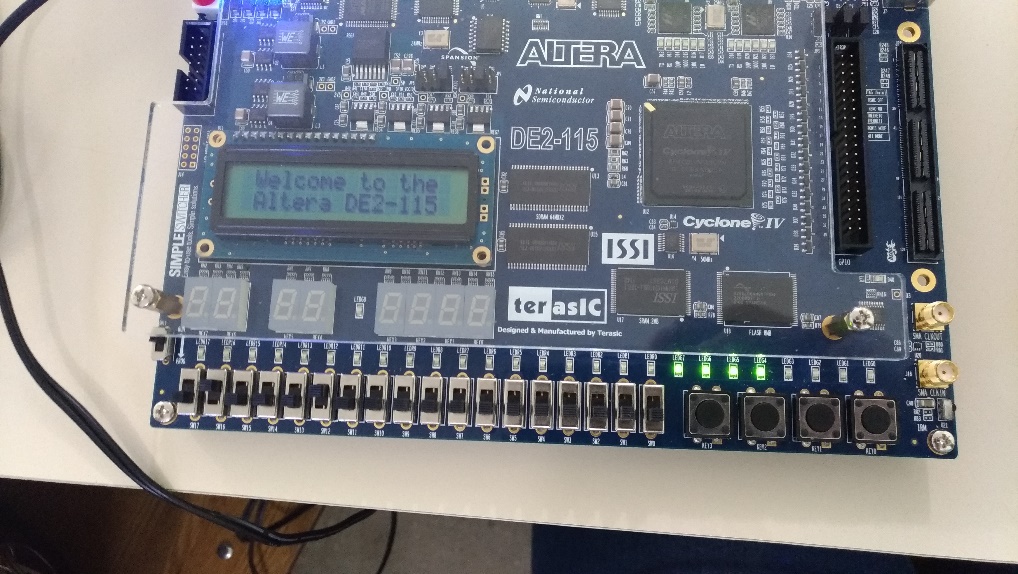
* Exemplo da conversão do número 2:



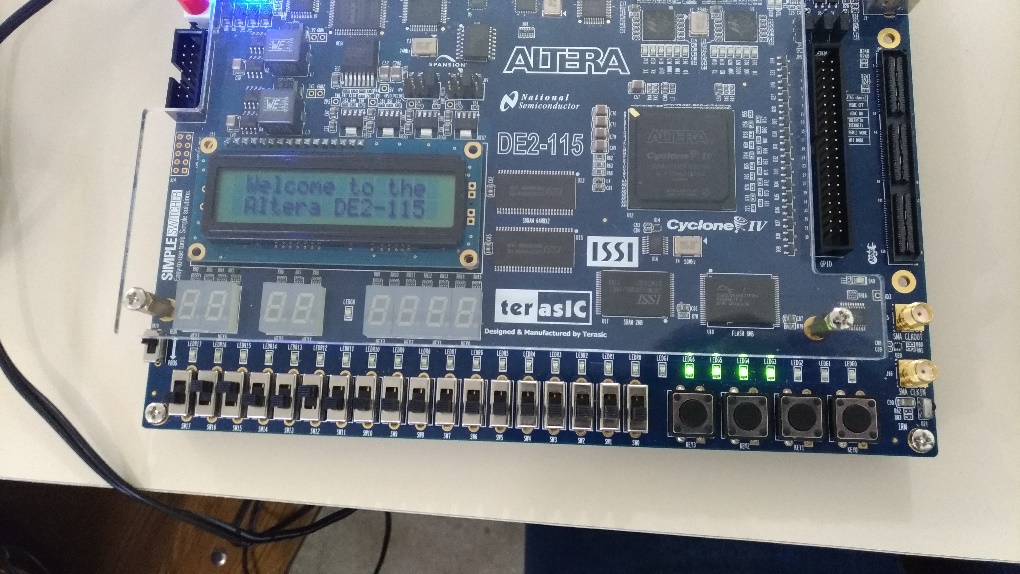
* Exemplo da codificação do número 3:



* Exemplo da codificação do número 4:



* Exemplo da codificação do número 5:
* Exemplo de codificação do número 6:



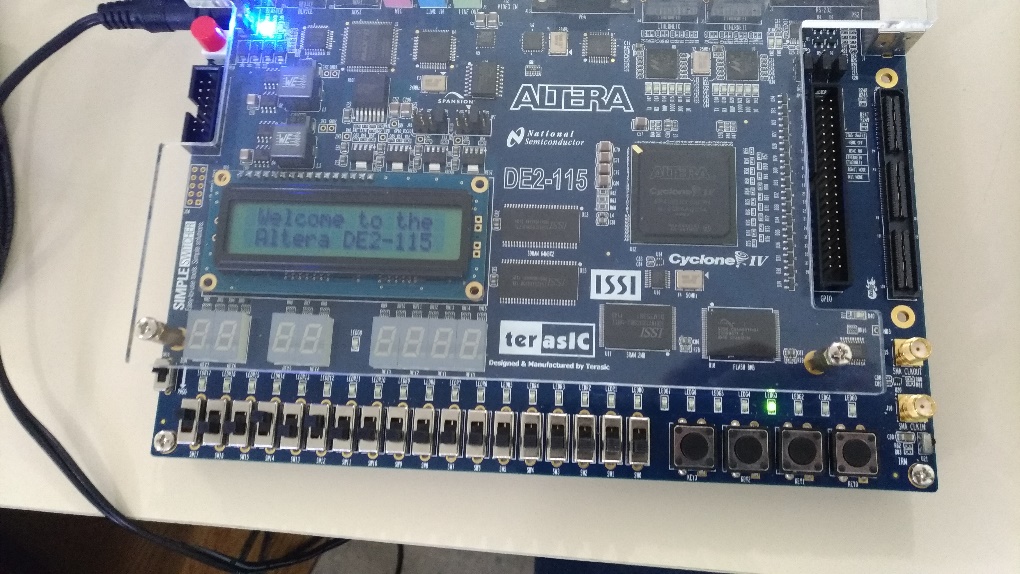
* Exemplo de codificação do número 7:

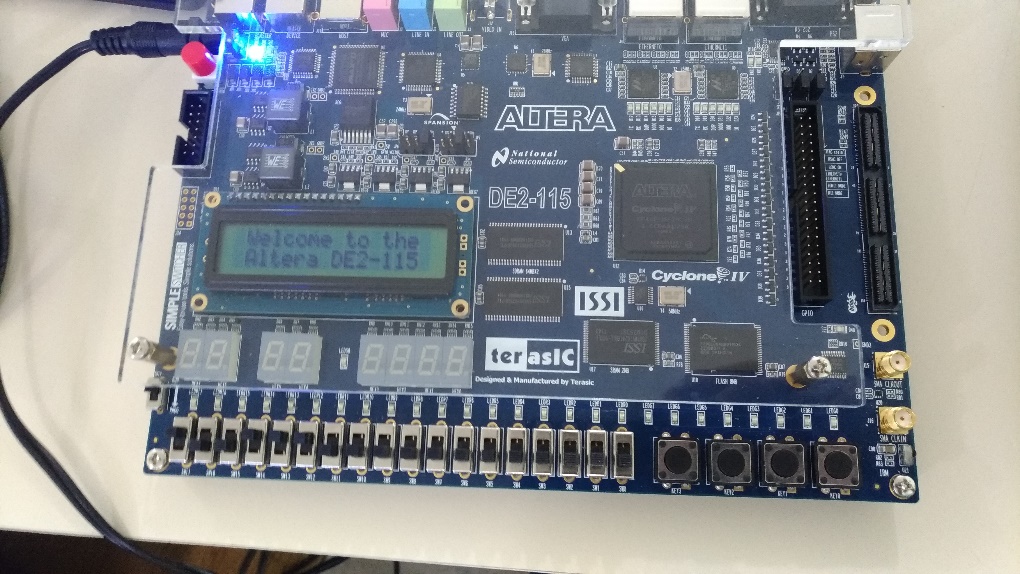


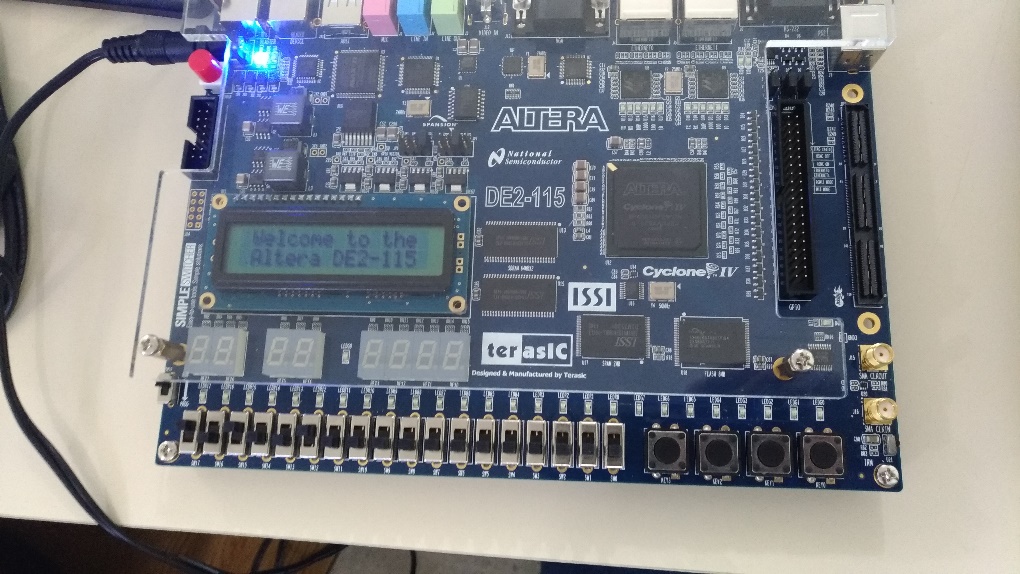
* Exemplo de codificação do número 8:



* Exemplo de codificação do número 9:



* Exemplo de *reset* ativado:
* Exemplo de entrada inválida(superior a 9):



1. **Agradecimentos**

Agradecimento especial para o monitor Lucas Duarte. Todas as aulas de monitoria foram de extrema relevância para a realização do trabalho com êxito.