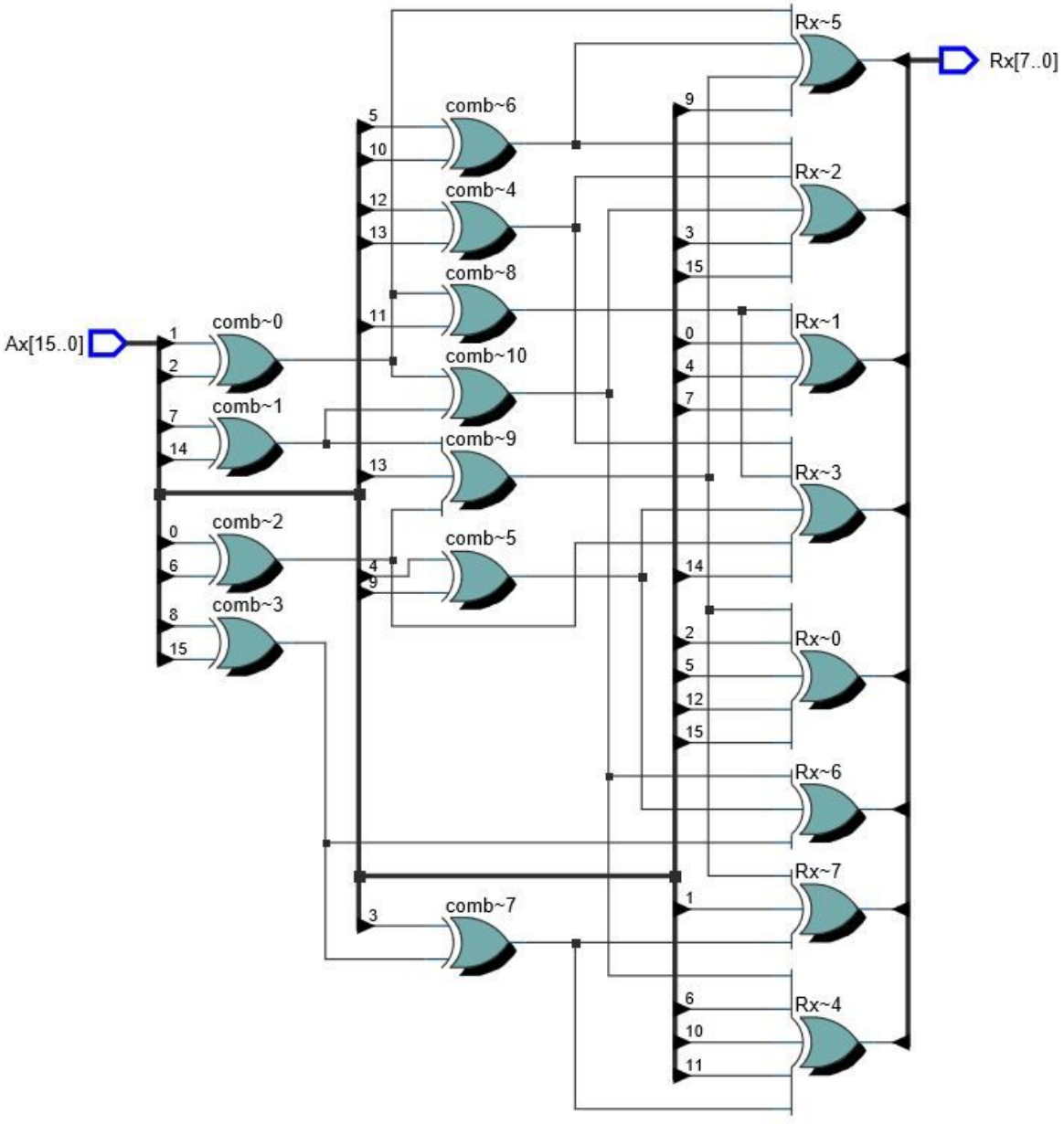
CRC_T1G1

Miguel Maia 76434

Diogo Candeias 68454

Implementação do Encoder



Esta implementação encontra-se na pasta *Polinomial Implementation - Just 2xorbit1.*

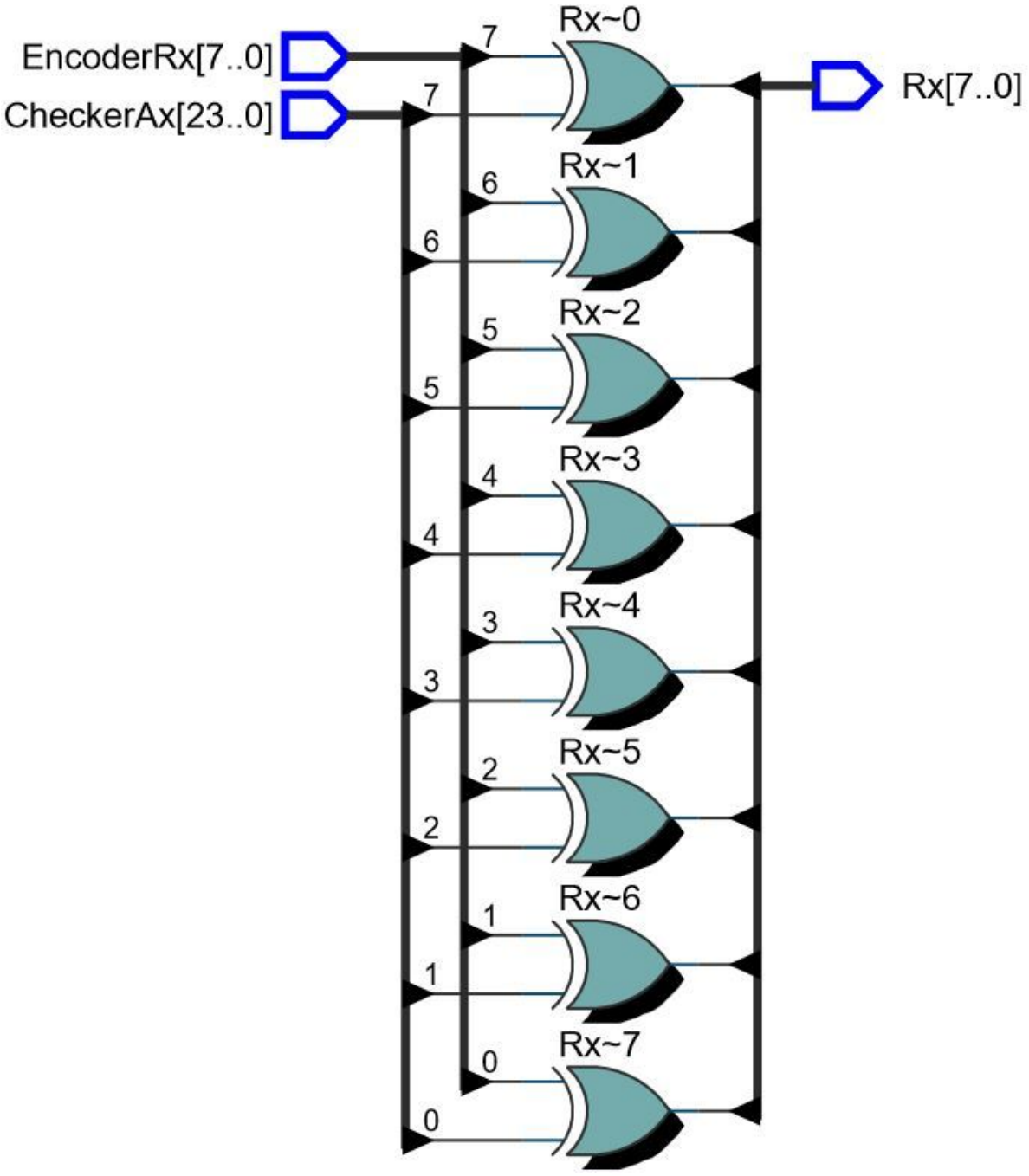
Esta implementação baseou-se na divisão de polinômios. Com a informação apresentada pelo professor na aula, seguiu-se a mesma lógica e conseguiu-se obter as expressões gerais para cada bit de saída do encoder:

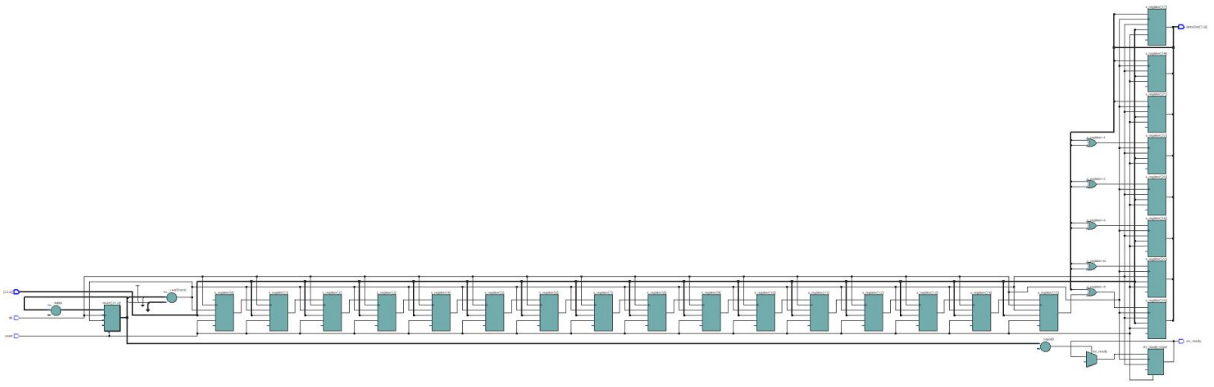
Nota: considere-se x7 o MSB da saída do encoder e o x0 o LSB.
x7 -> a15 xor a14 xor a13 xor a12 xor a7 xor a6 xor a5 xor a2 xor a0
x6 -> a11 xor a7 xor a4 xor a2 xor a1 xor a0
x5 -> a15 xor a14 xor a13 xor a12 xor a10 xor a7 xor a5 xor a3 xor a2 xor a1
x4 -> a14 xor a13 xor a12 xor a11 xor a9 xor a6 xor a4 xor a2 xor a1 xor a0
x3 -> a15 xor a14 xor a11 xor a10 xor a8 xor a7 xor a6 xor a3 xor a2 xor a1
x2 -> a14 xor a13 xor a10 xor a9 xor a7 xor a6 xor a5 xor a2 xor a1 xor a0
x1 -> a15 xor a14 xor a9 xor a8 xor a7 xor a4 xor a2 xor a1
x0 -> a15 xor a13 xor a8 xor a7 xor a6 xor a3 xor a1 xor a0

Este foi o ponto de partida. De seguida, através dum script e manualmente encontrou-se o número de repetições de pares. Posteriormente, substitui-se esses mesmos pares por um sinal cujo resultado é o ou-exclusivo do par, e repetiu-se o processo até não ser possível estabelecer pares.

Implementação do checker

Para o checker, o processo, consistiu em fazer um ou-exclusivo entre os *8 bits do resultado do encoder* com *os 8 bits menos significativos da entrada do checker*. Caso o resultado fosse 0, significaria que a mensagem transmitida não continha erros. Caso houvesse um 1, indicaria o bit do erro da mesma.
No caso de se pretender saber apenas se houve ou não erros, poderia-se fazer uma operação nor (Or negado) entre todos os bits do resultado do exor entre os *8 bits do resultado do encoder* com *os 8 bits menos significativos da entrada do checker.*


EncoderRx[7..0]
CheckerAx[23..0]
Rx[7..0]
Rx~0
Rx~1
Rx~2
Rx~3
Rx~4
Rx~5
Rx~6
Rx~7
7
7
6
6
5
5
4
4
3
3
2
2
1
1
0
0



Nota: Ver imagem na pasta LFSR.

A implementação do checker com 1 shift-register com 24 bits de entrada inicializado com os valores de A e R concatenados, consiste num simples shift à esquerda, dos valores posicionados nos 4º, 6º e 8º bits mais significativos (posições em que B tem os bits a 0) fazendo um XOR com o bit mais significativo nos restantes casos, dentro dos 9 bits mais significativos(tamanho de B).

O shift conclui a sua tarefa após 16 ciclos de relógio, após o último reset, altura em que para de “shiftar” e coloca a flag “crc_ready” a 1 sinalizando que o cálculo finalizou.

Nota: Existem na pasta *Extra* outras soluções que foram implementadas ao longo do desenvolvimento do projecto, no entanto conseguiu-se soluções melhores, tendo sido estas o ponto de partida. Como foram pedidas apenas 4 transparências decidiu-se não mencionar as mesmas neste relatório para não ficar demasiado extenso.