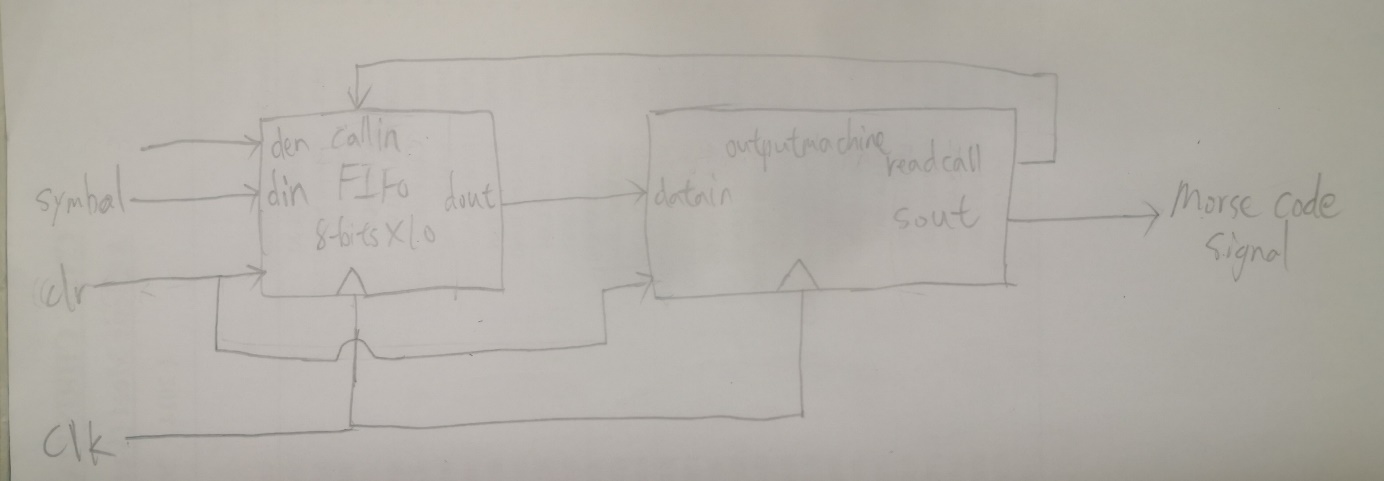
ELEC3342 Digital System Design Final Assignment Report

UID: 3035470967 Name: Leung Lap Yung

Design of my mc encoder:



The design mainly consisted of two components: **FIFO** and **outputmchine**.

**FIFO**:

Memory is used to store an array of input symbols(std\_logic\_vector). Head/Tail act as two pointers. Head tracks the last latest input symbol while Tail tracks the oldest symbol in the array. Whenever there are read-in or output operations, Head/Tail increments by 1. The FIFO takes in a callin signal which is an instruction to order FIFO to output one symbol given by the ouputmachine.

**Outputmachine**:

The function of the outputmachine is to map whatever symbols to std\_logic outputs. A counter is used to track the output stage of each symbol. Also, a readcall is made by outputmachine which corresponds to callin for FIFO as communication between them.

**Assumption**:

As the user would only input the next sentence after the previous sentence is printed out, the operations are synchronous. Based on this fact, my design need not deal with the problem when outputmachine needs to decide when to tell FIFO to start to output the next sentence.

The logic of my design:

1. FIFO reads a sentence until full-stop. Then FIFO starts to output one symbol to outputmachine.
2. outputmachine takes the symbol and outputs it. Upon the last std\_logic output, it makes a readcall to FIFO and FIFO would push the next symbol.
3. The recurrence goes on until a full stop is outputted, the outputmachine no longer makes readcall. Then the loop is distorted and the whole system is reset to initial condition for next sentence.

Why a request- feedback system is used:

Instead of using FSM, for simplicity of the implementation, I used 2 modules to synchronize the output of each character. That is, FIFO waits for the command from outputmachine to push the next symbol, the outputmachine makes request upon finishing a whole output of one character. This avoids using an FSM which usually consisted of many states and thus the codes would become complex.

However, the request- feedback system needs to deal with the latency problemThere is a 2-cycles delay from outputmachine makes a request and by the time it receives a new symbol.

The delay can be perfectly fitted into the system as there are letter gap and word gap between any 2 valid symbols.

For letter gap, simply add one more made delay before outputting the next symbol. So, a letter gap consists of 2 cycle delay from request- feedback system and 1 added delay. As default output is ‘0’, a letter gap can be represented by 3 ‘0’s.

For word gap, simply treat it as a symbol which output a ‘0’. So, a word gap consists of a letter gap, then a ‘0’, then a letter gap before next symbol. A total of 7 ‘0’ represents a word gap

Process of outputting a symbol:

An integer counter is used to track the so call output stage of each symbol.

The counter is initialised to 0 whenever a symbol is detected at outputmachine and incremented by 1 after each clock cycle.

A ‘0’ is outputted at counter value 0, as to solve the letter/word gap problem stated above.

For example, to output A, which is dot dash in Morse code, the outputs for each counter values are:

0: ‘0’

1: ‘1’ --dot

2: ‘0’

3: ‘1’

4: ‘1’

5: ‘1’ --dash

The counter stops here with the outputmachine. The outputmachine makes an push request to FIFO and waits for the next symbol