MA machine – draft 2

Introduction:

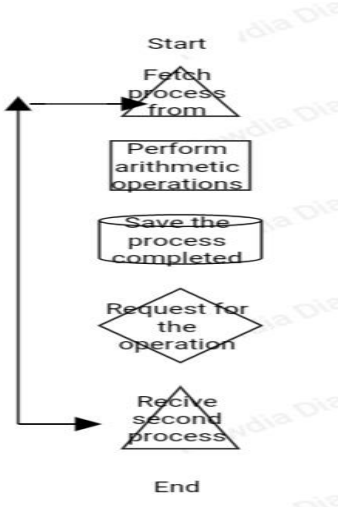
I have been working on a research myself to get more out of computer's after the announcement of IBM making 2 nano meter chips. I knew after this we will have to depend on quantum computers and it will still take a while so just like GPU is used to do the calculations for graphics instead of CPU I have decided to make a machine that will help CPU and we will get results much faster.

Let's start with a simple example let's say you are at home and someone told you to make rice now for example you have a assistant with you and you said him/her about the order and told him to make every possible dish of rice, fried rice, shezwan rice, Italian rice etc. Now the rice assistant made won't be useful now take this same scenario and let's go to a restaurant and now the customer said I didn't order simple rice I ordered shezwan now you don't have to make it the assistant already has it. Exactly how my MA machine would work.

Overview:

I will give you a overview of the MA machine in this paragraph by explaining you what it is now we have a ready queue the processes that are ready to be executed by are on the wait as CPU is already working on P1 and next P2 depending on the solution. Now P1 arrived at CPU I will take the copy of that process and perform every possible operation addition, subtraction, multiplication, division, log, anti log. And I will keep that operation on the memory of the MA machine now if it's not needed it will be deleted as the ready queue is empty.

Flow Chart:



Algorithm:

1. Fetch process from CPU (copy)
2. Perform all arithmetic and non arithmetic operations
3. Save it in the memory up until requested.
4. As soon as the request comes from CPU it will be directly passed to it.
5. Receive second process and repeat the process.
6. Delete existing process from the MA machine memory (reset) as soon as ready queue is empty.
7. End process.

It is a 64 bit machine to calculate all possible operations that are thrown to it to avoid errors.

Real World Example:

1. The calculations about the DNA or Fluid is pretty complex if I have the possible calvulatioifor example if the CPU is calculating the speed of fluid differentiating with respect to time the machine will keep the track of process and perform all possible operations.
2. The video streaming of higher quality and games of higher quality needs operations very similar for example let's just say CPU is calculating the simple grass asset now that asset is a type that has to be rendered again and again now if my machine is present it will take care of that process by taking a copy of grass operation from CPU and start calculation on every possible operation the screen will be filled with grass already same with mountains and what not.

Dual Output and Distributor:

1

.

. Mid → 1. 0.1 – 0.5 , 0.5 – 0.9

.

0

As we have 0 and 1 as the information into the CPU I have created partition of these 0 and 1 and distributed them in 2 different sections 1.) 0.1 – 0.5 2.) 0.5 – 0.9.

I have also created a formula to convert this MA codes (0.1 – 0.9) to binary and vice versa and created a algorithm for it too.

Formula to Convert Binary to MA Codes:

$$\sum_{n=0}^{1} \boldsymbol{B\ Rn = B}$$

This formula is the summation that is summing from 0 to 1 on the variable Rn.

Rn = Random Numbers between 0 and 1 and summation of them to get B

B (L.H.S) = Binary Number Indicator to stay within this binary number given.

Eg: 011(B) = 3 and MA code = 0.1 + 0.5 + 0.9 + 0.4 + 0.8 + 0.3 = 3. This is the reason there is a B (R.H.S) so whenever summation of Rn reaches the value it will be equal to that binary number.

Algorithm to convert Binary to MA Code:

1. Take the clock input assign it to a variable s <= 1
2. Now take that variable and assign it as a indicator to stay within this range.
3. Take a summation from 0 to 1 and assign it to Rn variable.
4. Rn variable has a condition of:

```
If s < 0:
    Wait
If s > 1:
    Wait
If s == Rn:
    Proceed
```

This algorithm is the perfect way to demonstrate my formula for converting Binanry to MA.

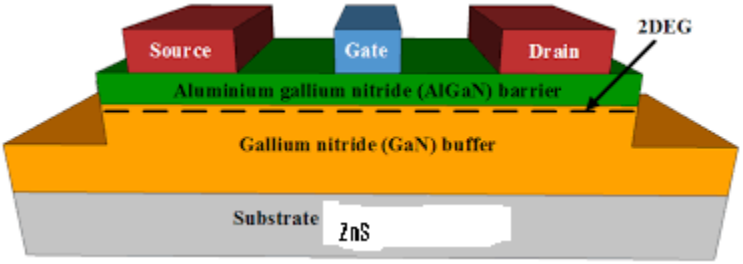
Dual Output:

At first I was building something that will be of dual output like 0 and 1 at the same time but the experiment failed because no matter how hard and forcefully I make it go through my machine the output was not desired at all. So now that we have the convertor that can give the outputs between 0 and 1 I don't need a dual output

Distributor:

I have created three logic gates together the are called TMP Gates. Now they stand for TLow, Midlow, PreciseLow. These gates are made up of HEMT transistor that will have a P type doped with ZnS (Zinc Sulphide) this will allow me to have a larger Band Gap between the NPN layer and a substrate.

Daigram for HEMT with ZnS:



TLow: This is a transistor with a input of 0 or 1 and outputs the lower voltages like 0.1 to 0.5. This will be useful for my machine as it is not going to work traditionally like 0s and 1s so we need such transistors to bypass that difficulty.

Symbol:

Truth Table/ Characteristics Table:

Clock	Input = A	Y = $\sum_0^1 Rn$
1	0,1	0.1
1	0,1	0.2
1	0,1	0.3
1	0,1	0.4
1	0,1	0.5

MLow: This is a transistor with a input of 0 or 1 and outputs the lower voltages like 0.5 – 0.9. This will be useful for my machine as it is not going to work traditionally like 0s and 1s so we need such transistors to bypass that difficulty.

Symbol:

Truth Table/ Characteristics Table:

Clock	Input = B	Y = $\sum_0^1 Rn$
1	**0,1**	**0.51**
1	**0,1**	**0.61**
1	**0,1**	**0.71**
1	**0,1**	**0.81**
1	**0,1**	**0.91**

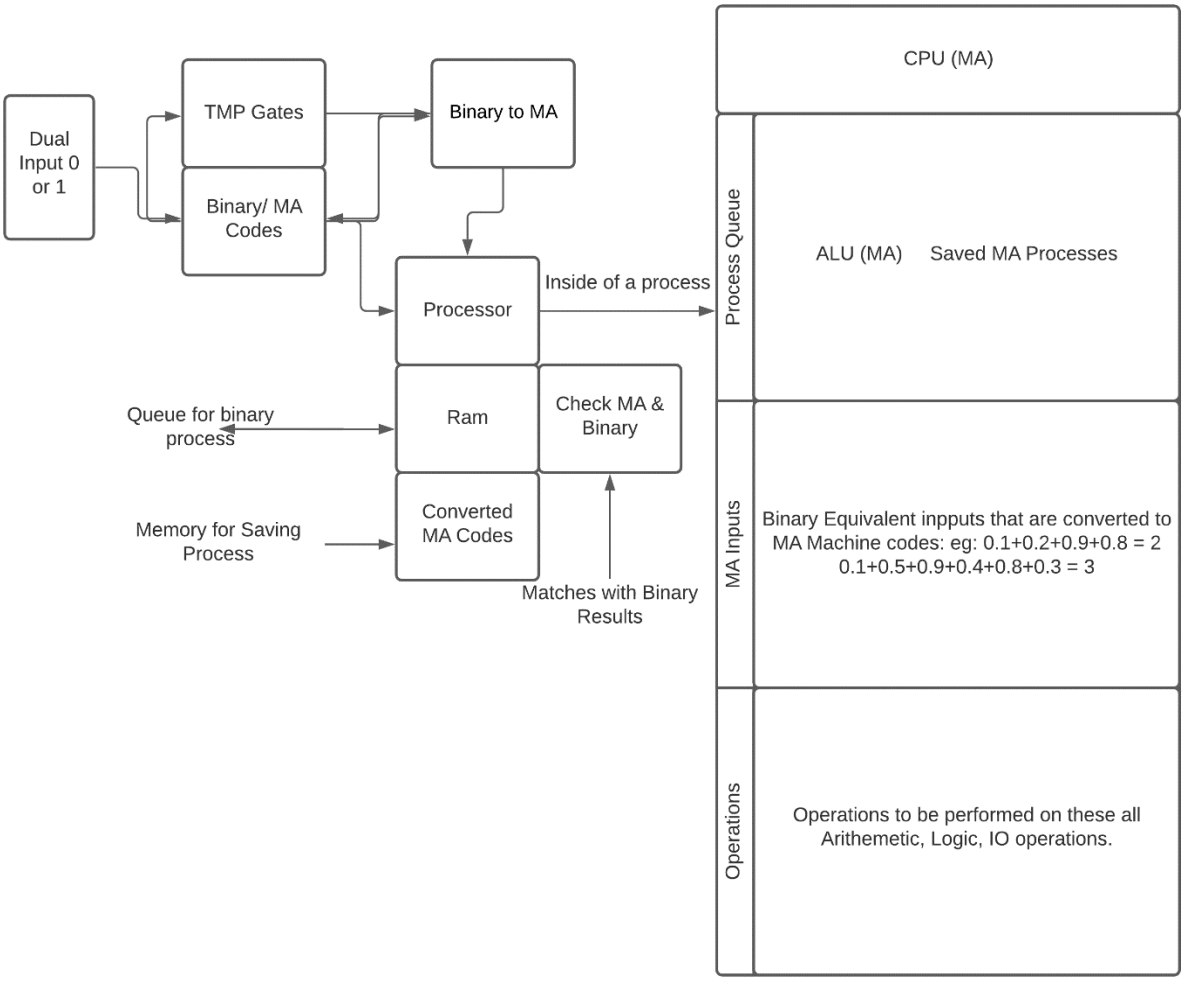
PLow: This is a transistor with a input of 0 or 1 and outputs the lower voltages like 0.1111…. – 0.99999... This will be useful for my machine as it is not going to work traditionally like 0s and 1s so we need such transistors to bypass that difficulty.

Symbol:

Truth Table/ Characteristics Table:

Clock	Input = A	Y = $\sum_0^1 Rn$
1	**0,1**	**0.12112132**
1	**0,1**	**0.25487678**
1	**0,1**	**0.33698712**
1	**0,1**	**0.44421264**
1	**0,1**	**0.57454529**
1	**0,1**	**0.65684154**
1	**0,1**	**0.73513542**
1	**0,1**	**0.83131535**
1	**0,1**	**0.94654131**
1	**0,1**	

MA Machine Architecture:

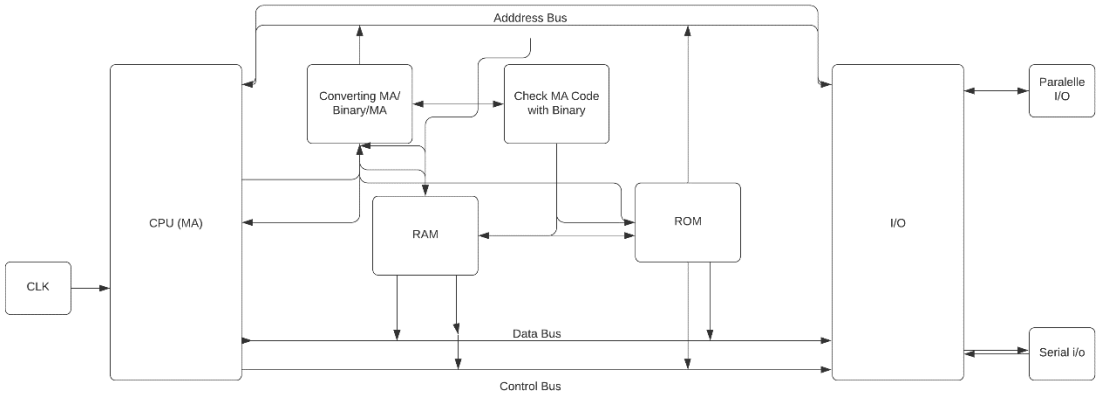


Architecture Explained:

1. There are total of7 components as you can see in the above figure. As you can see there is ojnly one arrow pointing towards the right diagram and its for processors and that's the inner working of how a MA machine process will look like there is still working yet to be explained.
2. The block TMP Gates/ Circuit are the gate that are mentioned on the TMP gates section above this so I can get the desired output of my liking.
3. Below that is binary or MA codes in this block I will save the binary codes provided to me by the information and I will send it to the block of Convertor that is Binanry to MA convertor.
4. Once converted I will send it to the memory to save these process the block below RAM will be responsible for saving MA codes so when CPU asks for the necessary codes it will be ready in my ready queue.

5. The block named check MA & Binary is nothing but what it suggests it will check if the results match with the RAM that has queue for CPU itself.
6. The inside of a Processor that is MA inputs, Process Queue, Operations are:
    - **Process Queue:** It has ALU for MA and the saved MA processes that is connection to the converted MA codes.
    - **MA Inputs:** All the inputs that are all the necessary codes given by the block will be present here and the information or permission calculate these to get desired output will be given by Operations that is next to it.
    - **Operations:** This block will contain all the information on what are we supposed to do with thses inputs of MA next to it like arithmetic, logic, or I/O it will send in information on if you should add of manipulate the given MA codes.

Block Daigram of MA Machine:



Block Daigram explained:

1. This block diagram is pretty similar to that of a normal microprocessor because it will be working like the microprocessor or else how am I supposed to make it useful if I don't have I/O ports, Paralell I/O, or even serial port that's most important.
2. I have added the necessary blocks from the Architecture to this Block diagram like converting MA/ Binanry, Check MA with Binary, so I can explain it.
3. As Address Bus, Data Bus, Control Bus are necessary to br connected with RAM & ROM the block of converting MA/ Binanry is also very crucial as it is the heart of my machine or else its just joke in the way of innovation.

I am a electronics engineering graduate in third year I have just worked on this and don’t know if it will be helpful I think it will be as it came from my thought experiment It might be way too wrong but I am trying my best to be a young researcher.

Thank you for reading.

A paper by **Usama Shamsuddin Thakur**