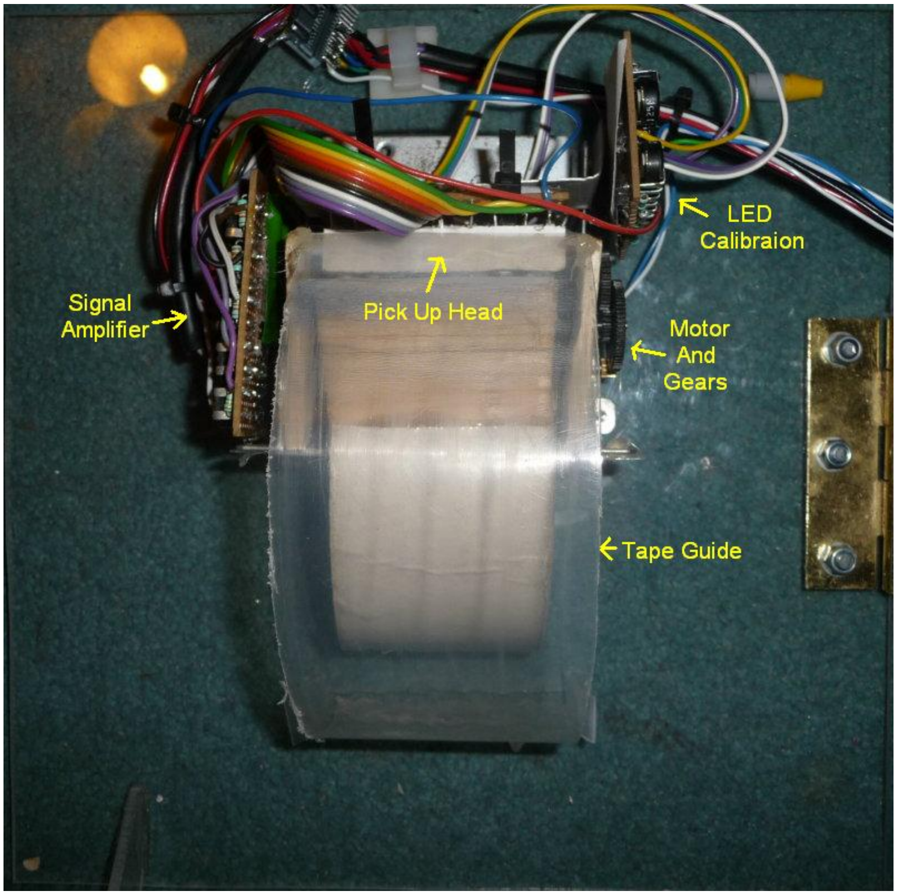
TIM

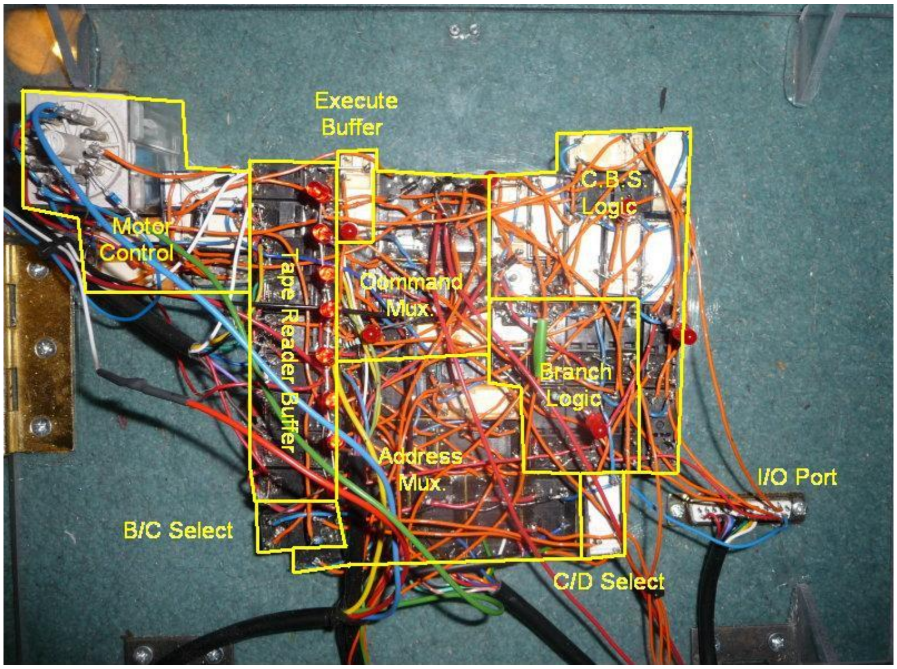
# GALLERY

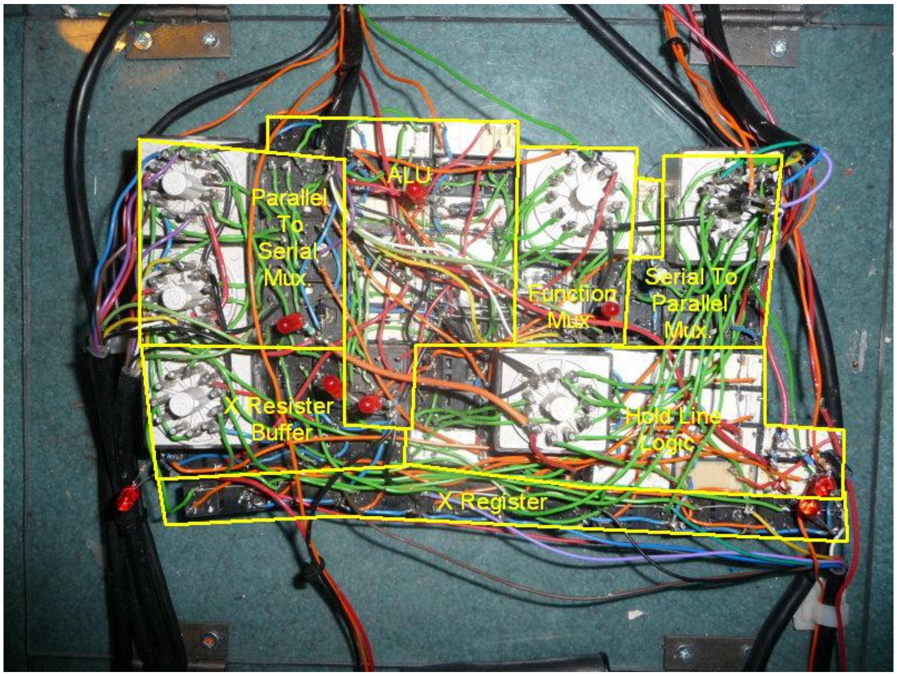
Back to TIM 8

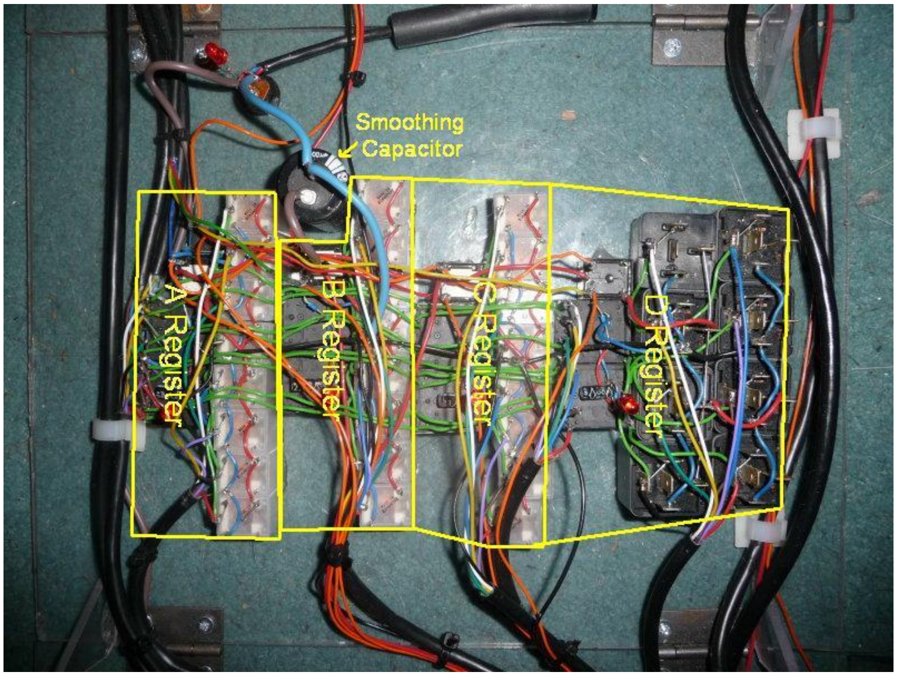
Here are some more detailed pics of the inner working's of TIM.

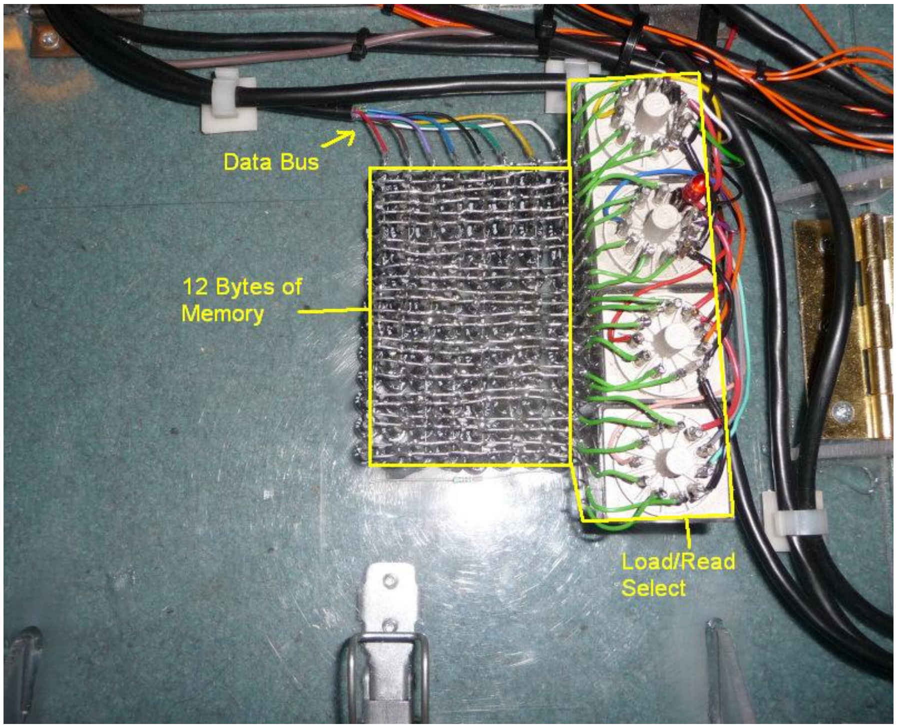
Below are pictures of each panel of the of TIM, each with lables to give you an idea of where everything is. ('Mux' just stands for Multiplexer)

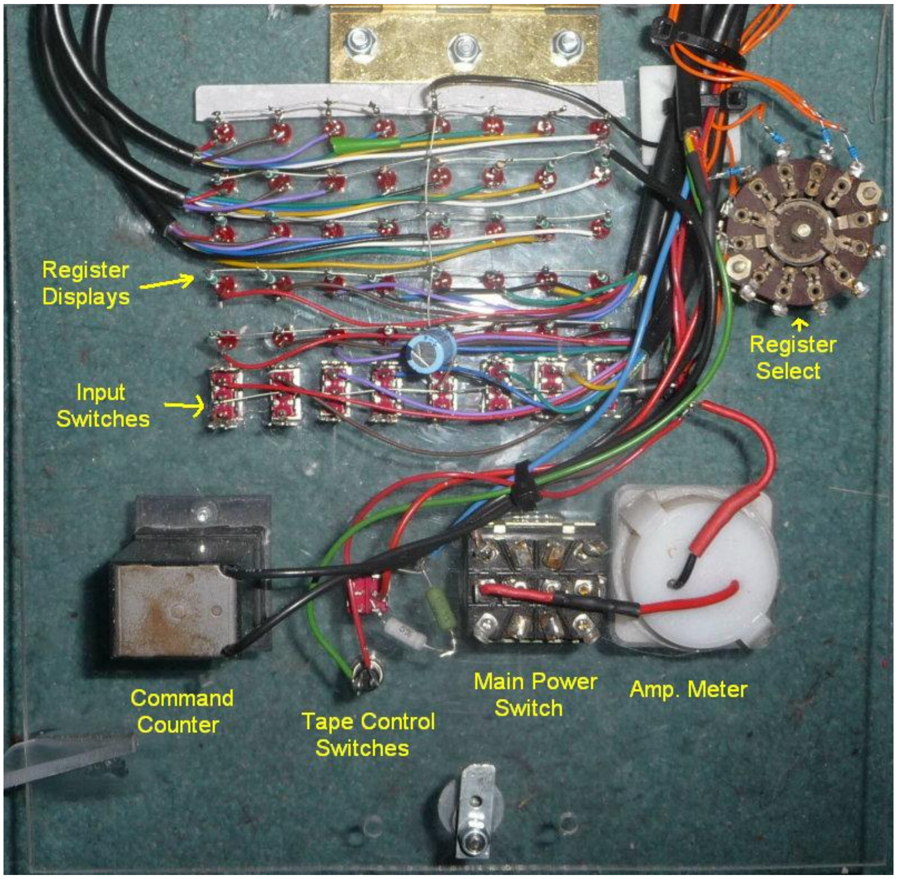



Execute
Buffer
C.B.S.
Logic
Motor
Control
Tape Reader Buffer
Command
Mux.
Branch
Logic
Address
Mux.
I/O Port
B/C Select
C/D Select

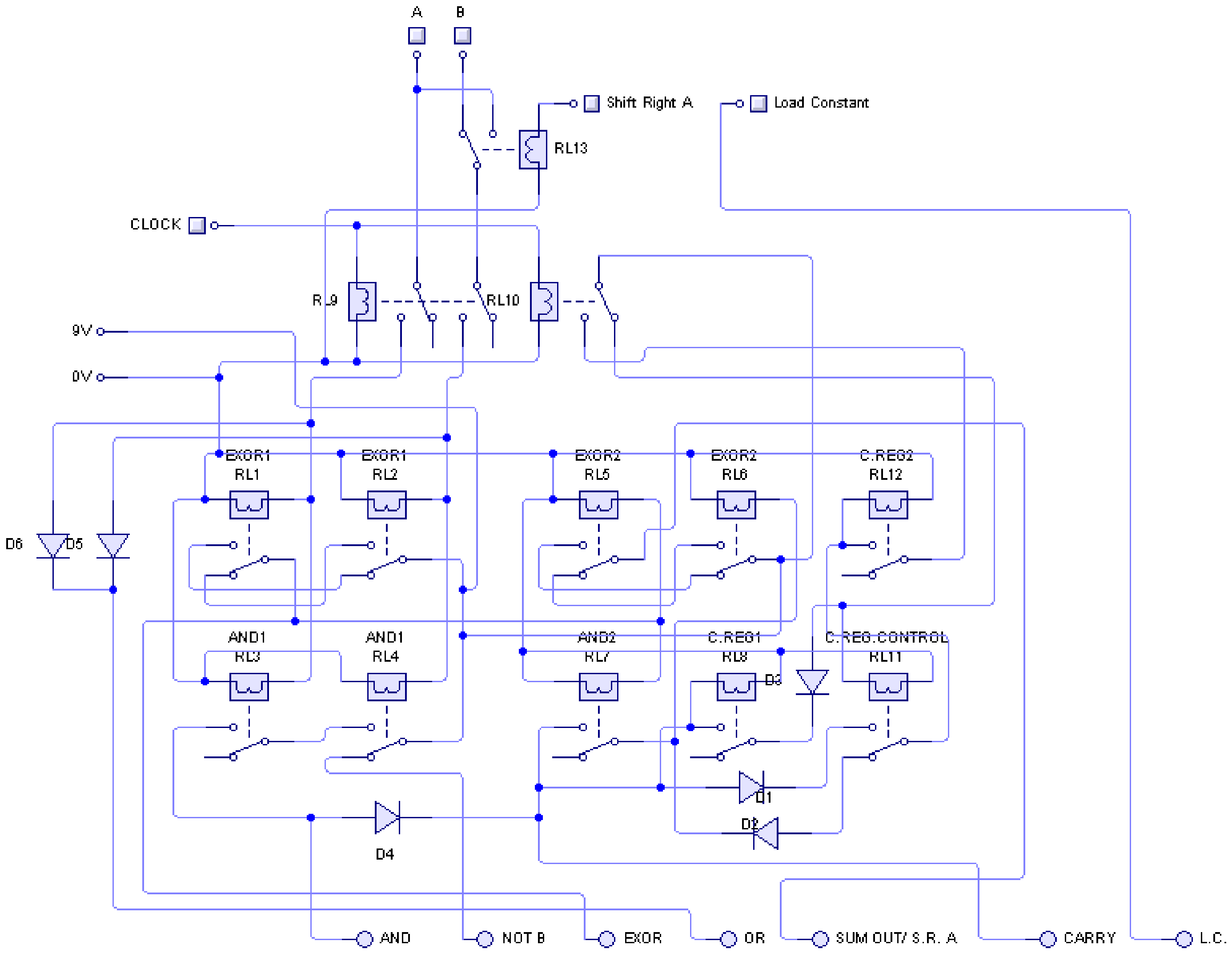

ALU
Parallel
To
Serial
Mux.
Function
Mux
Serial To
Parallel
Mux.
X Resister
Buffer
Hold Line
Logic
X Register


Smoothing Capacitor
A Register
B Register
C Register
D Register

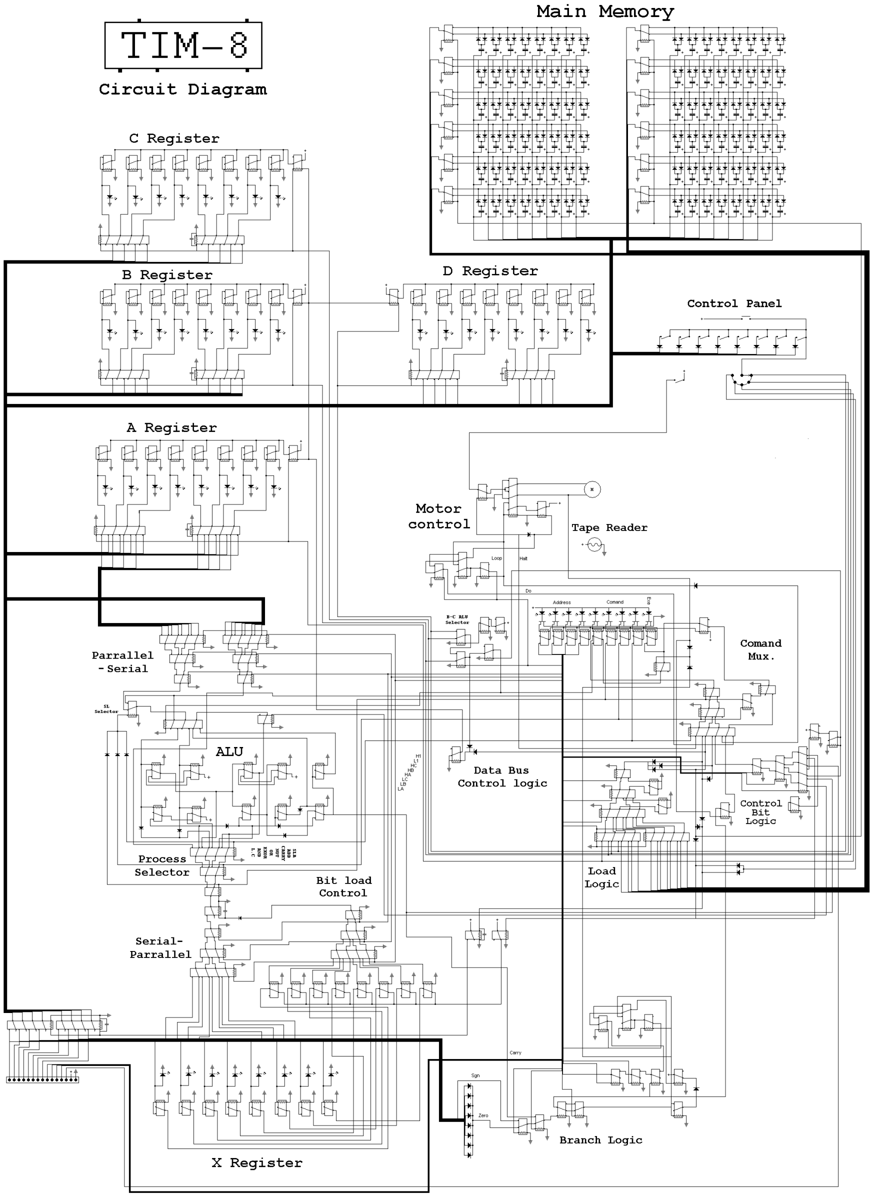

Data Bus
12 Bytes of Memory
Load/Read Select



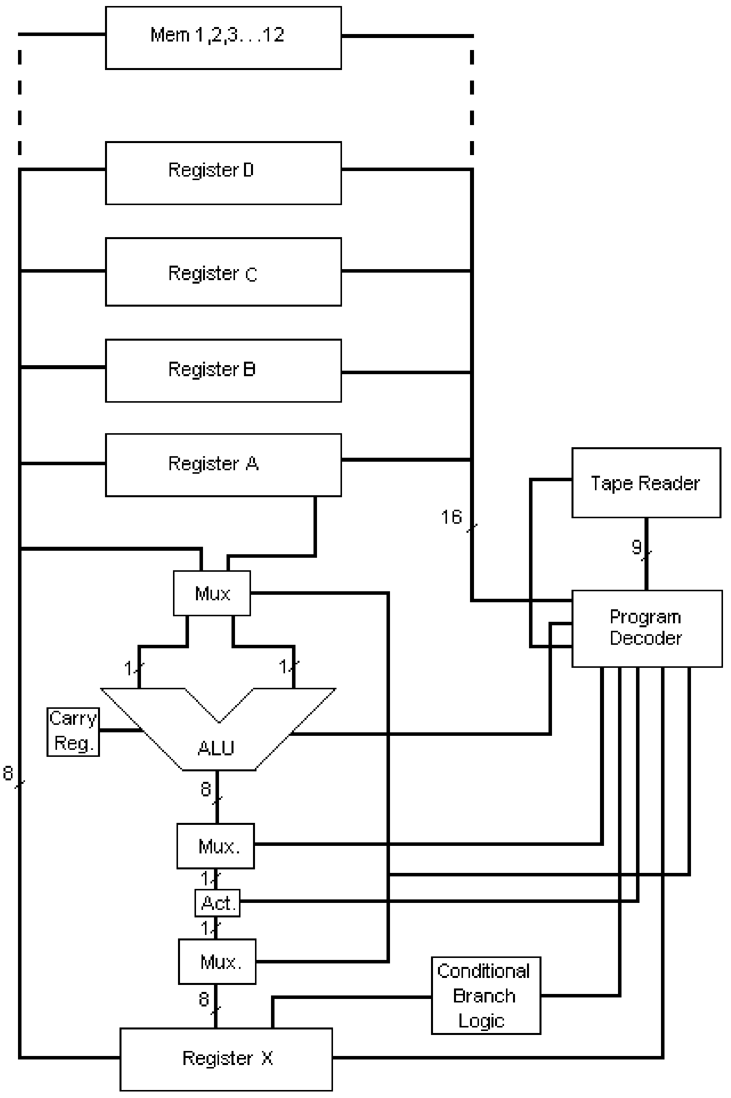
This is a picture of the actual circuit diagram for the ALU, showing all 8 function outputs. It was drawn up in Circuit Wizard, which was where the design for this ALU was finalized:



Here is the finished design of TIM 8. It is mostly accurate, appart from a few minor changes made during debugging. Sorry about it being so disorganised, but when I started it was not supposed to get this big!



And this shows the general architecture of the whole system:



Here is the full list of T++ commands:

```
Do (1)
Loop (1)
Halt
Load to *
Load from *
Carry=0
Carry=1
Enable A
Disable A
Enable D
Disable D
Clear X
Act

If x<>0
If x=0
If x-
If x+
If carry=0
If carry=1
Endif

SL 0-7 *
ADD 0-7 *
CARRY 0-7 *
```

NOT 0-7 *
OR 0-7 *
EXOR 0-7 *
AND 0-7 *
LC 0-7 *

FUNCTIONS:

INPUT *
SUBTRACT * (from A)
INC *

(And any Binary Instruction
From BLT)

Any where where there is a *, either a memory locartion or register name is required. All ALU processes requiring 2 inputs take data from the A register and the register specified by the *

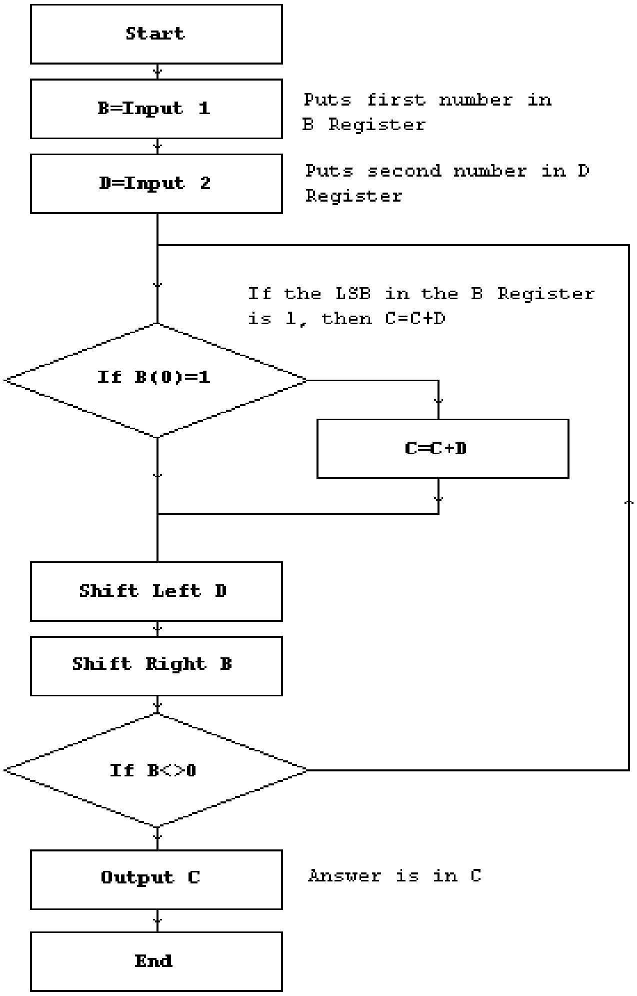
The image below demonstrates how binary multiplication works in T++:

# BINARY MULTIPLICATION

```
(D Reg.)     Top no.   101
(B Reg.) Bottom no.    x11
                       101
                      1010
(C Reg.)     Answer   1111
```

If the LSB on the bottom number is a 1, then the top number is added to the answer. The bottom number is then shifted right, and the top number is shifted left. The process is repeated until the bottom number is equal to 0 (LSB stands for Least Significant Bit or the bit on the right hand side.)

**Program Flowchart**


Start
B=Input 1
Puts first number in B Register
D=Input 2
Puts second number in D Register
If the LSB in the B Register is 1, then C=C+D
If B(0)=1
C=C+D
Shift Left D
Shift Right B
If B<>0
Output C
Answer is in C
End

**Program In T++**

**(Any ALU op Is performed on data from A Reg. and the reg. specified by the letter at the end.)**

```
Input B          Input B

Input D          Input D

Do               Repeat

Clear X          If B(0)=1
Load To A
OR 0-0 B   (moves LSB from B to X)
If X<>0

Load From D
Load To A        C=C+D
carry=0
Add 0-7 C
Load to C

endif            Endif

Carry=0          Shift Left D
Enable D
Sh 0-7 D
Disable D
Load to D

Carry=0          Shift Right B
sh 7-0 b
Load to B

If X<>0          Until B=0
loop
endif
```

Back to TIM 8