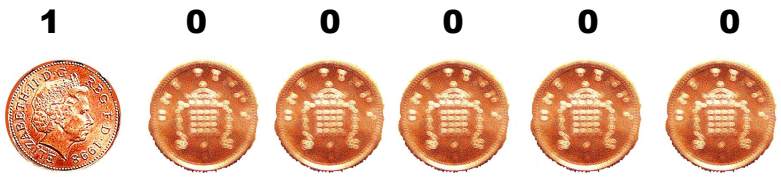
Size Does Matter Part Two

Binary

"Binary" means of two parts, e.g. 0 or 1, heads or tails, yes or no, true or false, the choice between two alternatives. If pennies are used with 1 represented by a head and 0 represented by a tail, 100000 would appear as shown in Figure 1.

Figure 1

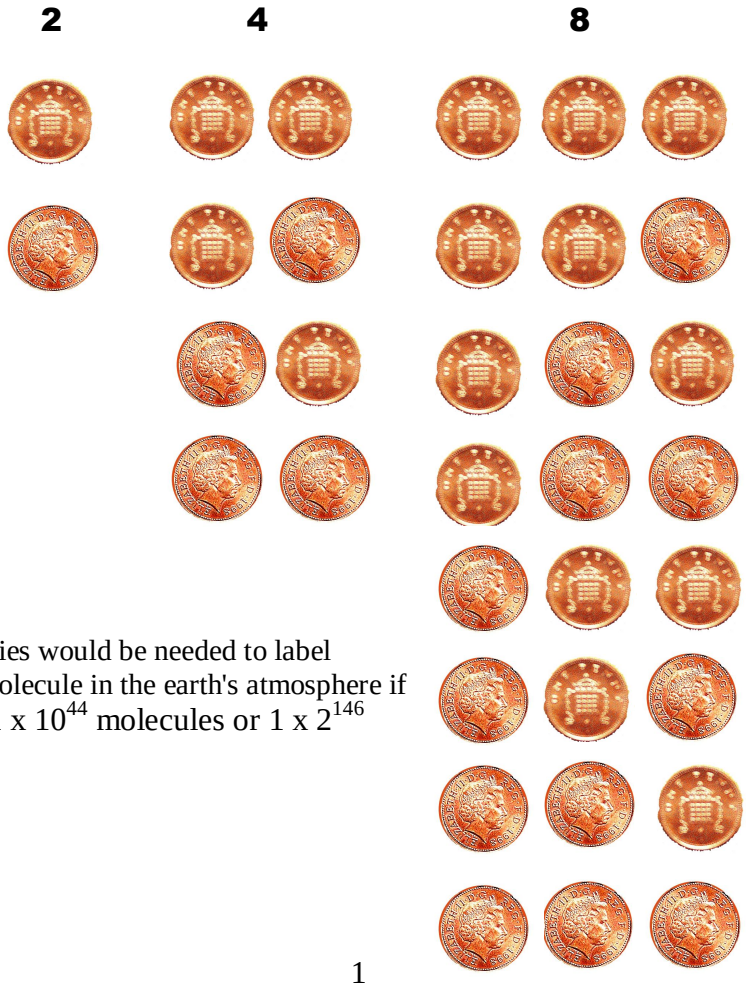


Tail up penny codes 0,
Head up penny codes 1

Figure 2 shows the penny patterns for one penny, two pennies, and three pennies. The number of penny patterns doubles each time, doubling means multiplying by two and incrementing the power of two by one.

$$2^1 \rightarrow 2^2 \rightarrow 2^3 \rightarrow 2^4 \rightarrow 2^5 \rightarrow 2^6 \rightarrow 2^7$$

Figure 2



How many pennies would be needed to label uniquely each molecule in the earth's atmosphere if there are about 1×10^{44} molecules or 1×2^{146} molecules?

Answer: 146

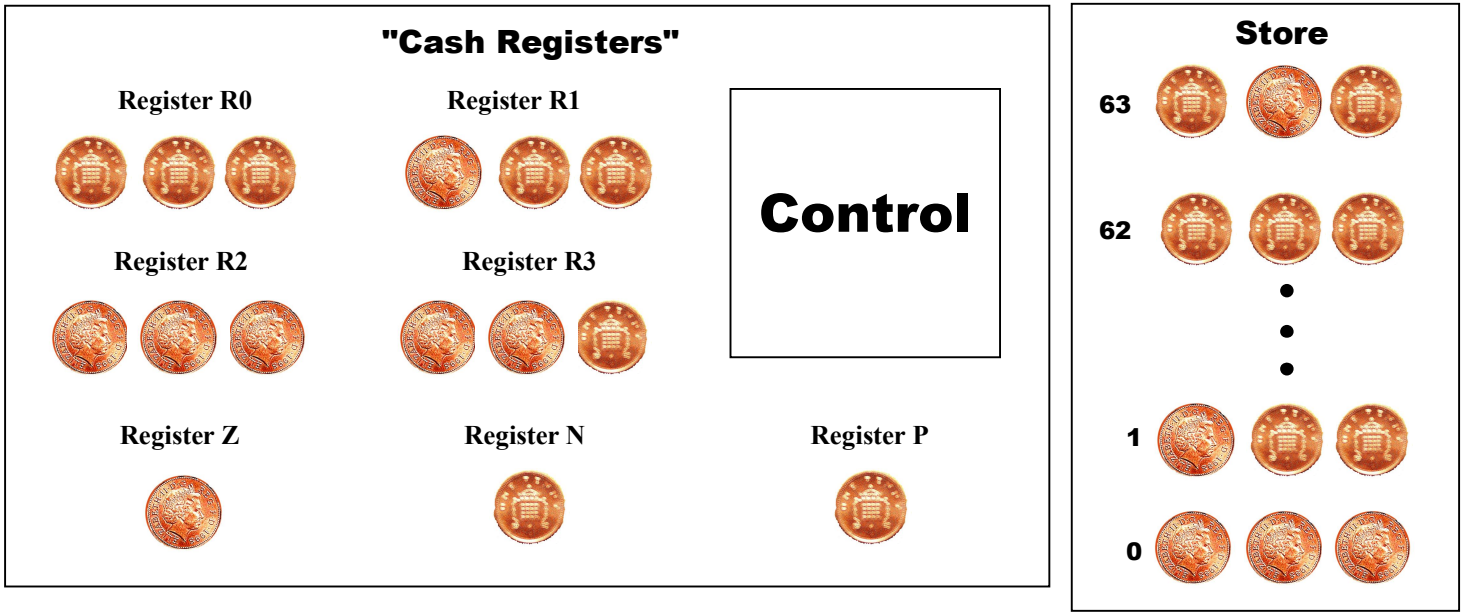
© Dr K R Bond 2011

The Penny Binary Computer

Students struggle to understand the role of machine registers and machine operations. The **Penny Binary Computer** is one approach that the author has used to introduce students to machine architecture and machine operation, successfully. It can also support a better understanding of why some algorithms take more time to process data than other algorithms. The machine steps to perform an operation are exposed for students to experience.

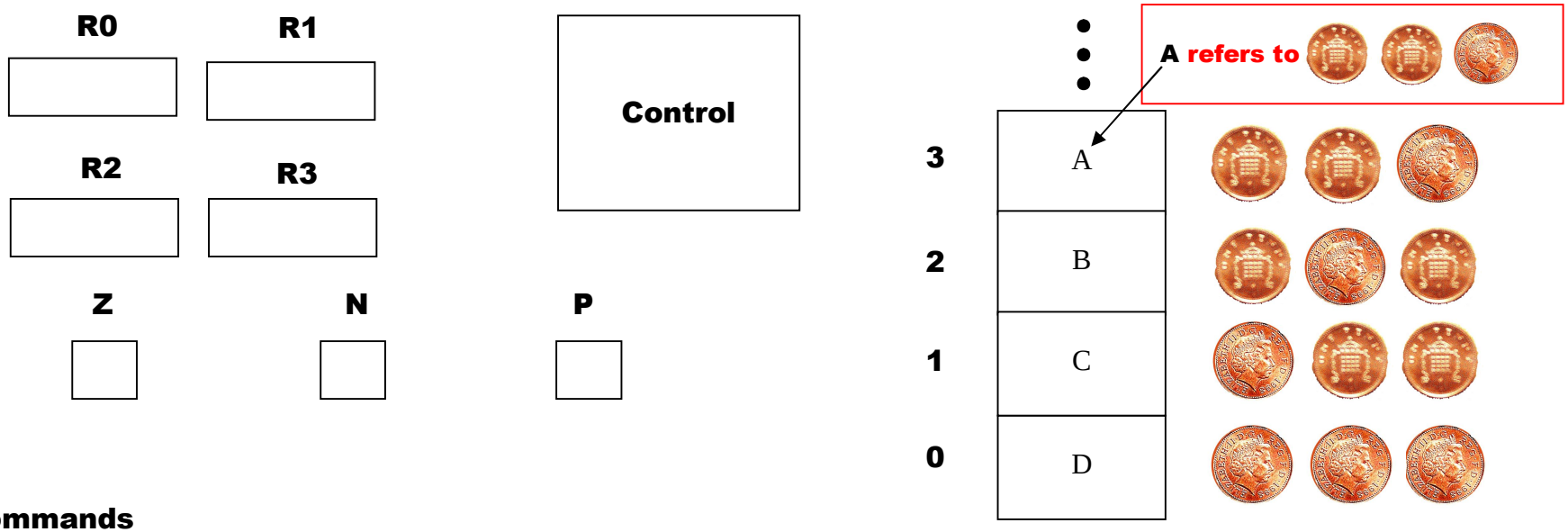
In the **Penny Binary Computer** pennies are moved and copied from a store of pennies to cash registers of pennies and vice versa. The store consists of **64** racks each of three pennies, cash registers each of three pennies and cash registers each of a single penny as shown in Figure 3. The cash registers are labelled with the names **R0**, **R1**, **R2**, **R3**, **Z**, **N** and **P**. The store racks are labelled **0**, **1**, **2**, **3**, **4**, etc. Figure 4 shows the registers without the pennies.

Figure 3



© Dr K R Bond 2011

Figure 4



Commands

The command **Move #0, R0** means copy **zero** into the register **R0**. In the penny computer, this means placing the penny pattern

 in the register **R0**.

The command **Move R1, R0** means copy **content of R1** into the register **R0**. In the penny computer, this means placing the penny pattern

 in the register **R0** when **R1** contains  .

The command **Move(R0), R3** means copy what is stored in memory location with address specified by content of **R0** into register **R3**, i.e. address is the penny pattern stored in **R0** 

© Dr K R Bond 2011

If the content of this memory location is  then the content of **R3** becomes 

The command **Compare #4, R0** performs a subtraction. **4** is subtracted from the content of **R0** and if the outcome is **zero**  is placed in the **Z** register otherwise  is placed in the **Z** register. If the outcome is **negative** then  is placed in the **N** register otherwise  is placed in the **N** register. If the outcome is **positive** (> 0) then  is placed in the **P** register otherwise  is placed in the **P** register. Register **N** indicates that the outcome of the last machine operation was negative.

The command **Add #1, R0** increments the content of **R0** by 1 so  in **R0** becomes 

in **R0**. The command **BranchOnNegativeResult** ***somelabel*** will transfer control to the statement labelled ***somelabel:*** if the **N flag** is set, i.e. 

The command **BranchOnResultZero** ***somelabel*** will transfer control to the statement labelled ***somelabel:*** if the **Z flag** is set, i.e. 

The command **JMP** ***somelabel*** will transfer control unconditionally to the statement labelled ***somelabel:***. The command **Halt** stops the execution.

Bubble Sort

The following program rearranges the penny patterns in memory locations **0, 1, 2, 3** so that the largest binary pattern value ends up in **3**, the next in **2**, the next in **1** and the least in **0**.

```
            Move #0, R0       /Store 0 in R0
                              / R0 represents NoOfPasses
StartNextPass:Add #1, R0     /Increment R0 by 1

            Compare #4, R0    /R0 - 4
            BranchOnResultZero LabelStop

            Move #0, R1 /Store 0 into R1      / R1 points to the
                  /first of the pair of locations to be compared
```

R0 

Z  **N**  **P** 

R0 

R1 

© Dr K R Bond 2011

```
LoopStart:      Move (R1), R3 /Copy contents of the first of the paired
                              /memory locations (R0) into R3

                Move (R1 + 1), R4 /Copy contents of the second of the
                      /paired memory locations (R1 + 1) into R4

                Compare R4, R3   /R3 - R4 and set the appropriate flag
                BranchOnNegativeResult LabelIncMem

                Move R4, (R1)    /Copy contents of R4 into memory
                      /location at the address indicated by contents
                      /of R1, e.g. 0
                Move R3, (R1 + 1)     /Copy contents of R3 into memory
                      /location at the address indicated by contents
                      /of (R1 + 1), e.g. 1
LabelIncMem:    Add #1, R1  /Increment contents of R1 by 1
                Compare #4, R1  /R1 - 4 and set appropriate flag
                BranchOnResultZero StartNextPass
                Jmp LoopStart
LabelStop:      Halt
```

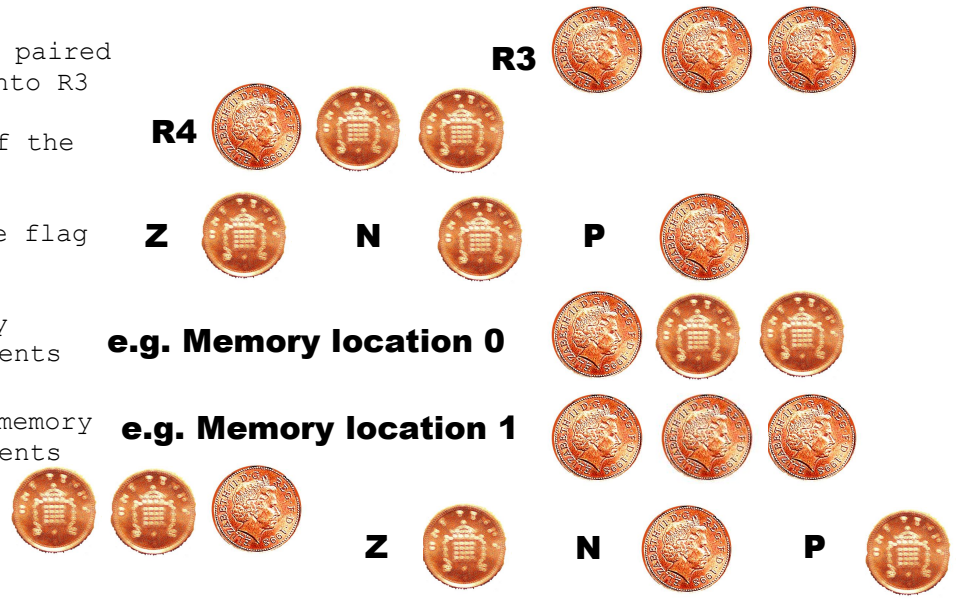
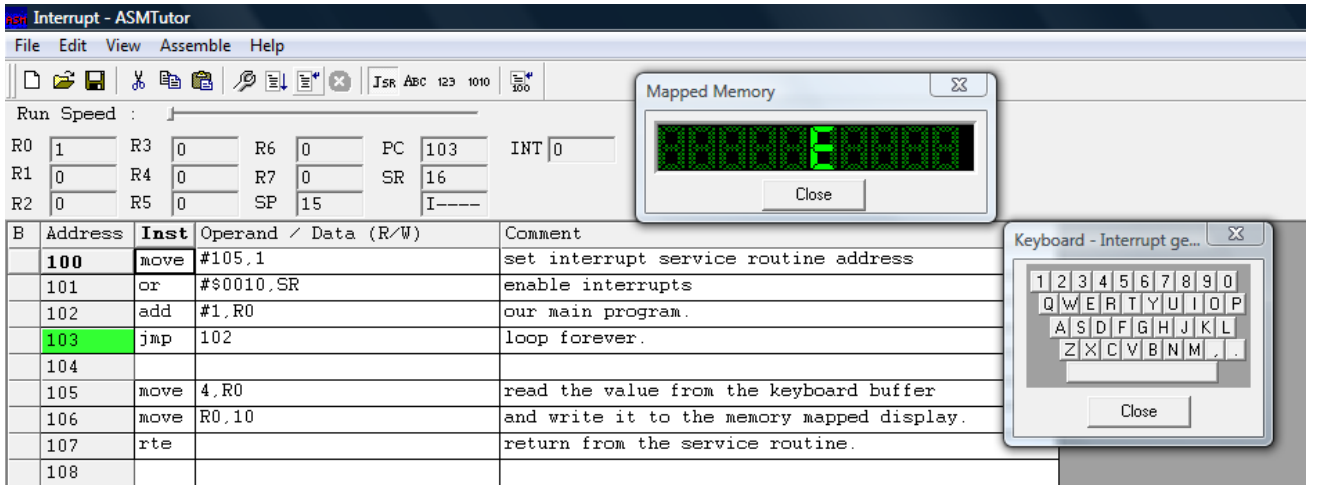


Figure 5

ASM Tutor

The Penny Binary Computer leads on to a machine simulator such as ASM Tutor. Figure 5 shows ASM Tutor executing an assembly language program which simulates accepting input from a keyboard and output to an alphanumeric display. The registers and their role in the execution of the program can be seen clearly. The PC or Program Counter points to the next instruction to be fetched and executed. ASM Tutor is available from Educational Computing Services Ltd.



© Dr K R Bond 2011