Computer Logic

Register

A register is a small but very fast storage location available to the CPU to store data temporarily. Registers found inside the CPU can be **general purpose registers**, that is, registers which are used by the CPY to hold temporary data and instructions. However, there are also **special purpose registers**. A special purpose register is one that has a specific control or data handling task to carry out. There are a number of special purpose registers within the CPU such as the Instruction Register, Program Counter and Accumulator.

Range of a register

Registers have a fixed size. Therefore, a register has a range of numbers that can be stored in it. The range of a register is described by giving the smallest and the largest number that can be stored in that particular register. The range of a register changes according to the size of the register. However, one has to consider if the numbers stored in the register are represented in unsigned binary or 2's complement.

Range of an 8-bit <u>unsigned binary</u> register:

Smallest number:	00000000_2	=	0_{10}
Largest number:	11111111_2	=	255_{10}

So, in an 8-bit unsigned binary register, one cannot store a number which is larger than 255 and smaller than 0.

Range of an 8-bit <u>2's complement</u> register:

Smallest number:	10000000_2	=	-128_{10}
Largest number:	01111111_2	=	127_{10}

So, in an 8-bit 2's complement register, one cannot store a number which is larger than 127 and smaller than -128.

Overflow Error

When the CPU is carrying out calculations, it could be that the result of a calculation is too large to fit in the register. If this happens, the answer stored in the register is incorrect and we call that an **overflow error**.

If we consider an 8-bit (unsigned binary) register, we know that the largest number that fits in the register is 255_{10}. So if the CPU tries to store the number 260_{10} in this register, it would not fit and therefore an overflow error occurs.

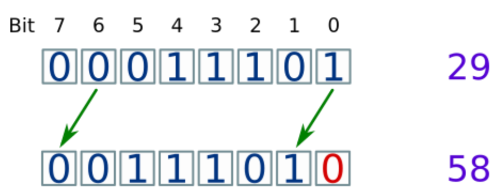
Arithmetic Left/Right Bit Shifts

When the contents of a register are shifted 1 bit to the left, the number stored in the register is doubled (x2). On the contrary, when the contents of a register are shifted 1 bit to the right, the number stored in the register is divided by 2 (÷2).

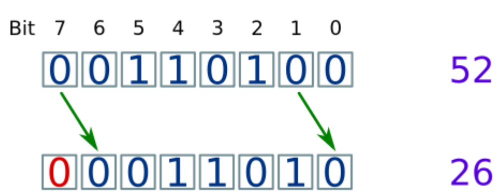
Example:

Consider the decimal number 29_{10} (00011101_2) as stored in an 8-bit unsigned binary register as shown below.

1-bit shift to the left: $29_{10} \times 2 = 58_{10}$



1-bit shift to the right: $52_{10} \div 2 = 26_{10}$



Number of combinations possible

The number of combinations of binary codes that one could create depends on the number of bits in the code. For example, if only 2 bits are available, 4 combinations of binary codes can be created: 00, 01, 10 and 11. If 3 bits are available, the number of combinations increases to 8: 000,001,010,011,100,101,110,111.

This relationship between the number of combinations and number of bits available can be expressed using the following formula:

$$2^n = k$$

Where n = the number of bits available while k = the number of combinations that could be created.