DIGITAL CONTROL CONCEPTS

This information is intended to supplement the instrumentation & controls lectures with additional information on the digital computer application for controls with implementation details.

Lecture Topics

1. Digital Logic Overview
2. Decimal Numbering System
3. Binary Numbering System
4. Octal Numbering System
5. Hexadecimal Numbering System
6. Computer Use of Numbering Systems
7. Input Subsystem
8. Computer Memory
9. Central Processing Unit
10. Output Subsystem

1. DIGITAL LOGIC OVERVIEW

- A logic decision can easily be displayed by changing the particular ***state of a device*** from say energized to de-energized.

- Imagine a list of questions that can be answered unequivocally by a ***yes*** or ***no*** answer. We could allocate the ***yes answer*** (or state) to the ***energized state*** and the ***no answer*** (or state) to the ***de-energized state***.

- As we only have two choices here (yes or no), this system can be classed as a ***binary*** system.

- A simple electric circuit with an ***on/off*** switch can be used to signal the binary ***yes*** or ***no*** condition by lighting a lamp in the circuit.

- The yes state or ***energized state*** could have the ***bulb illuminated.***

- The no state or ***de-energized*** state could have the ***lamp extinguished***.

- And so an observer could determine the ***answer decision*** by monitoring the bulb status.

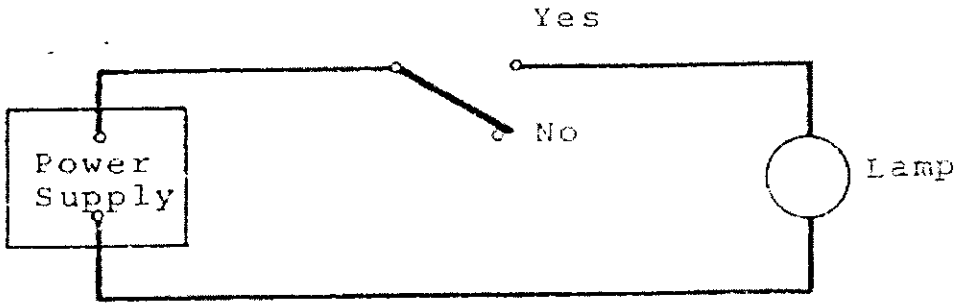

Figure 1. Binary Lamp State used to indicate yes or no condiiton

1. DIGITAL LOGIC OVERVIEW....continued

- Since this simple system has only two states, we can arbitrarily assign the numerical value of ***one*** (1) to the ***energized state*** and the value of ***zero*** (0) to the ***de-energized state***.

- A base two or ***binary number system*** is required to allow a computer to operate as a decision making (yes or no?) device.

- The smallest storage unit in the digital computer is one **bi**nary digi**t** which is contracted to **bit**.

- The bit will be displayed as either a ***0*** or a ***1*** in the binary machine.

- Such a simple logic system consisting of these two states can be efficiently and conveniently utilized in computers to make ***continual binary logic decisions*** which can then be used for ***control purposes***.

NUMBERING SYSTEMS

2. The Decimal Numbering System

- The most common numbering system used is the ***base 10*** or decimal system.

- For example, the number 124 can be written as follows:

$(\underline{1} \times 10^2) + (\underline{2} \times 10^1) + (\underline{4} \times 10^0) = \underline{124}$ Base 10

- By knowing the ***sum*** of the assigned values for ***hundreds***, ***tens*** and ***ones***, we are able to determine the total value of 124 (which we already know because we are familiar with this system).

- It is worthwile to consider a specific value (i.e. 999) so that we can consider what happens when we add one to the number.

$(\underline{9} \times 10^2) + (\underline{9} \times 10^1) + (\underline{9} \times 10^0) = \underline{999}$ Base 10

- Now let's ***add one*** to this number:

$999 + 1 = (\underline{9} \times 10^2) + (\underline{9} \times 10^1) + (\underline{9+1} \times 10^0)$

$= (\underline{9} \times 10^2) + (\underline{9} \times 10^1) + (\underline{10} \times 10^0)$

- But note that (10×10^0) is equal to (1×10^1) and so the ***ones*** value should be ***shifted upward*** to the ***tens*** position.

2. Decimal Numbering System.....Continued

- ....and so the ***ones*** value should be ***shifted upward*** to the ***tens*** position. , resulting in the following:

$(\underline{9} \times 10^2) + (\underline{9} \times 10^1) + (\underline{10} \times 10^0) = (\underline{9} \times 10^2) + (\underline{10} \times 10^1) + (\underline{0} \times 10^0)$

- Again, the ***tens*** value of $(\underline{10} \times 10^1)$ is equal to $(\underline{1} \times 10^2)$ and so the ***tens*** value should be ***shifted*** to the ***hundreds*** position, resulting in the following:

$(\underline{9} \times 10^2) + (\underline{10} \times 10^1) + (\underline{0} \times 10^0) = (\underline{10} \times 10^2) + (\underline{0} \times 10^1) + (\underline{0} \times 10^0)$

- Finally, the hundreds value of $(\underline{10} \times 10^2)$ is equal to $(\underline{1} \times 10^3)$ so that the ***hundreds*** value should be shifted to the ***thousands*** position. So after one final ***shift*** the sum of 999 + 1 becomes:

$(\underline{1} \times 10^3) + (\underline{0} \times 10^2) + (\underline{0} \times 10^1) + (\underline{0} \times 10^0) = \underline{\mathbf{1000}}$ Base 10

- It can be seen from this exercise that the ***largest number*** that can appear in a particular value position must be the ***base value*** (10 in this case) ***less one*** (so 9 in this case). If this largest number is exceeded, the numerical indication must ***shift*** upward to the next higher power position value.

3. The Binary Numbering System

- The binary or **base two** system used in computers will indicate the values for various powers of 2.

- The maximum digit allowed at any one position will be the ***base value*** (2 in this case) ***less one*** (or 1 in this case) so that only ***0*** or ***1*** will be utilized.

- Now to consider several ***binary examples*** to demonstrate this numbering system:

1. Represent **decimai 4** in binary code.
Decimal $4 = 2^2 = (\underline{1} \times 2^2) + (\underline{0} \times 2^1) + (\underline{0} \times 2^0) = \underline{100}$ Base 2

2. Represent **decimal 9** in binary code. Decimal $9 = 8 + 1$:
Decimal $8 + 1 = 2^3 + 2^0 = (\underline{1} \times 2^3) + (\underline{0} \times 2^2) + (\underline{0} \times 2^1) + (\underline{1} \times 2^0)$
$= \underline{1001}$ Base 2

3. Represent **decimal 49** in binary code. Decimal $49 = 32 + 16 + 1$:
Decimal $32 + 16 + 1 = 2^5 + 2^4 + 2^0$
$= (\underline{1} \times 2^5) + (\underline{1} \times 2^4) (\underline{0} \times 2^3) + (\underline{0} \times 2^2) + (\underline{0} \times 2^1) + (\underline{1} \times 2^0)$
$= \underline{110001}$ Base 2

3. The Binary Numbering System.....continued

- You can see from these examples that although the binary numbering system is simple, it may be easy for a person to make a mistake reading a large group of 1's and 0's in a row.

- As a result, the numers are usually grouped in an ***octal system*** (based on 8) or a ***hexadecimal system*** (based on 16).

- As before, the iargest single value in one position must be one less than the base number, so for ***octal the largest value is 7*** while for ***hexadecimal the largest value is 15***.

4. Octal Numbering System

- The largest numerical value (i.e. 7) in any one position in the octal system is represented as:

Decimal 7 = 4 + 2 + 1 = (1 x 2^2) + (1 x 2^1) + (1 x 2^0) = 111 Base 2

This becomes (0 x 8^1) + (7x 8^0) = 7 Octal

- As an example, convert ***Decimal 75*** to the ***Octal*** equivalent.

Decimal 75 = 64 + 8 + 3:

Decimal 64 + 8 + 3 = (1 x 8^2) + (1 x 8^1) + (3 x 8^0) = 113 Octal

- Note that the octal system is just a ***short hand way*** of writing the base two numbers. For the example;

Decimal 75 = 64 + 8 + 3 could be written in base two as follows:

64 + 8 + 3 = (shown on next line to allow enough room)
= (1 x 2^6) + (0 x 2^5) + (0 x 2^4) + (1 x 2^3) + (0 x 2^2) + (1 x 2^1) + (1 x 2^0)

- Grouping this value in sets of ***3 bits*** for maximum number seven at each position results in:

1 001 011 = 001 001 011 = 113 Octal (which we had found before)

5. Hexadecimal Numbering System

- The largest numerical value (i.e. 15) in one position in the hexadecimal system is represented as:

Decimal 15 = **8** + **4** + **2** + **1** = (**1** x 2^3) + (**1** x 2^2) + (**1** x 2^1) + (**1** x 2^0)
= **1111** Base 2

This becomes (0 x 16^1) + (**15**x 16^0)

- The hexadecimal number system corrsponds to the decimal system in the following manner:

Decimal: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Hexadec: 0 1 2 3 4 5 6 7 8 9 A B C D E F 10

so that **15** Decimal = **1111** Base 2 or **F** hexadecimal

- As an example, convert ***Decimal 279*** to the ***Hexadecimal*** equivalent.

Decimal 279 = 256 + 16 + 7
Decimal 279 + 16 + 7 = (**1** x 16^2) + (**1** x 16^1) + (**7** x 16^0)
= **117** Hexadecimal

- Note that the hexadecimal system is also just a short hand way of writing the base two numbers. For the example Decimal 279 = 256 + 16 + 7 could be written as base two as follows:

256 + 16 + 7 =
= (**1** x 2^8) + (**0** x 2^7) + (**0** x 2^6) + (**0** x 2^5) + (**1** x 2^4) + (**0** x 2^3) + (**1** x 2^2)
+ (**1** x 2^1) + (**1** x 2^0)

- Grouping in sets of ***four bits*** for maximum number fifteen at each position results in:

1 0001 0**111** = 0001 0001 0**111** = **117** Hexadecimal
(as we had shown before)

6. Computer Use of Numbering Systems

- Each ***task*** for the computer operation, such as ***add***, ***subtract***, ***multiply***, ***divide***, etc, can be assigned a unique number.

- The unique combination of 1's and 0's will always be recognized by the computer as the ***assigned task***.

- In this fashion the computer can execute an assigned program task by executing the ***coded numerical sequence instructions***.

- ***Descriptive labels*** can be applied to the resulting instruction set to faciliate use and checking by programmers or maintenance staff.

- For example, an ***add immediate*** instruction could be given the mnemonic ***ADDI*** and assigned the value 06120 Octal or 0C50 Hexadecimal.

- This means that anytime the computer sees the instruction 000 110 001 010 000, then an ***add immediate*** action will be completed.

- It is the responsibility of the programmer to ensure that ***data*** and ***instructions*** are always presented in the expected order and format.

General Computer Operation

- The computer can be considered as being made up of several ***interfacing sections***, each of which has a defined task.

These sections are:

- ***Input Subsystem***, (contact the field to read devices)
- ***Computer Memory***, (store designated data for future use)
- ***Central Processing Unit*** (CPU), (perform logic assessments)

and

- ***Output Subsystem.*** (contact the field to write to devices)

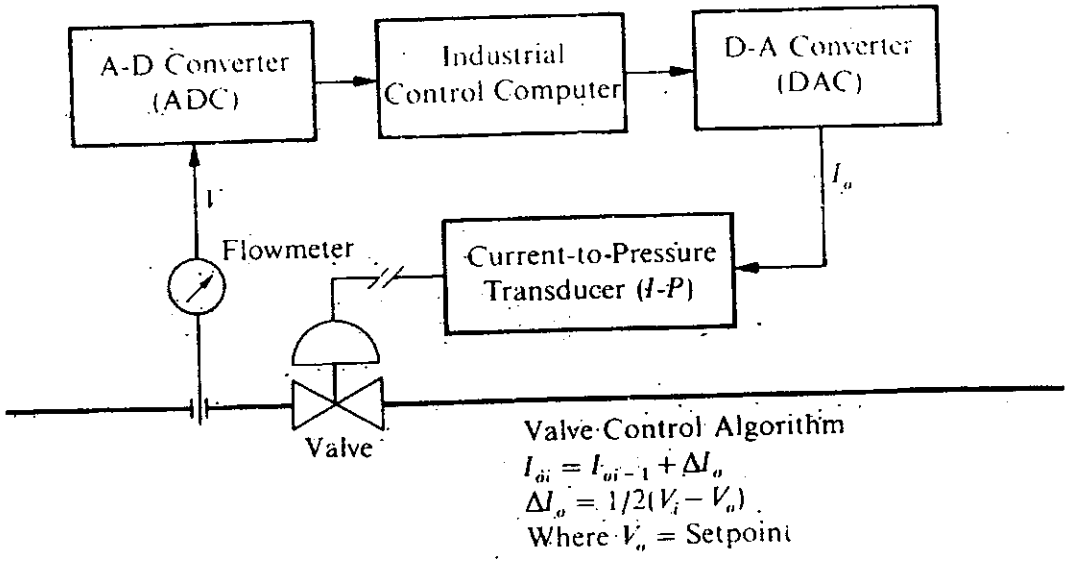

Figure 2. Digitized Analog Control Loop Diagram

7. Input Subsystem

- The Input Subsystem allows the computer to ***read devices*** so that actual ***plant data*** can be dynamically entered into the computer on a continual basis.

- Typical ***input devices*** may be ***analog to digital converters*** (ADC's), ***Digital Inputs*** (DI's), ***Keyboards***, ***Priority Interrupt Modules*** (PIM's), ***data links*** from other computers, ***reader devices*** (tape, disc, card,etc).

- The data input subsystem must be ***fast enough*** to track significant parametric changes to allow adequate control response times.

- The input subsystem must not ***burden*** (i.e. load down) the computer operation by the ***input read*** and ***conversion activities***.

- Sufficient ***barriers*** must also be provided to protect the computer from ***field generated faults*** and ***electrical disturbances***.

- Adequate ***redundancy*** of signals should be provided to allow high confidence assessment of signal ***rationality*** and ***validity*** as well as preventing a ***single failure*** from disabling the automatic computer control mode.

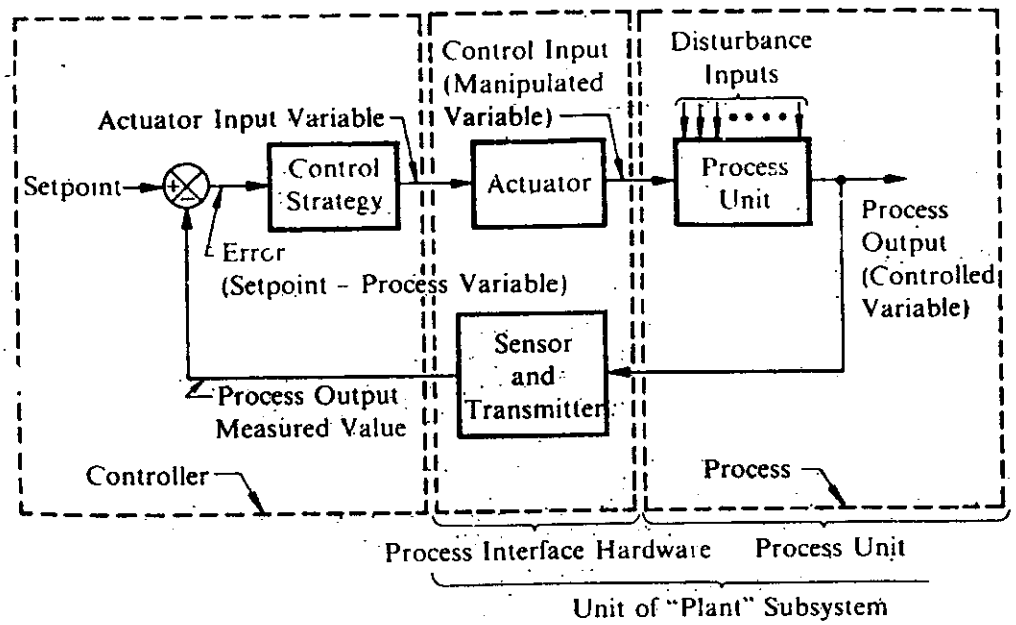

Figure #3 Generalized Plant Control Installation

8. Computer Memory

- This is the memory that the computer requires for ***hardware operation*** and ***program execution*** purposes.

- This memory is divided into Read Only Memory (**ROM**) and Random Access Memory (**RAM**).

- The ROM would contain the power up, ***bootstrap*** information necessary to start the computer and to conduct ***essential low level support*** functions in a continual, reliable manner.

- The RAM would contain the ***executive*** and application programs which monitors the plant status and makes corrective control decisions.

- RAM application programs are characteristically ***loaded from disc*** at power-up time once the computer system has been powered up and execution mode started (i.e booted up).

- RAM is an immediate or ***short term memory*** (for that execution cycle and perhaps some short term storage capability).

- For longer term data storage, the designated data could be ***stored to disc*** within allocated datafiles so that specific plant data over a requested calendar date and time could be called up for future review - this is usually referred to as a ***historical data storage*** and ***retrieval*** system (HDSR).

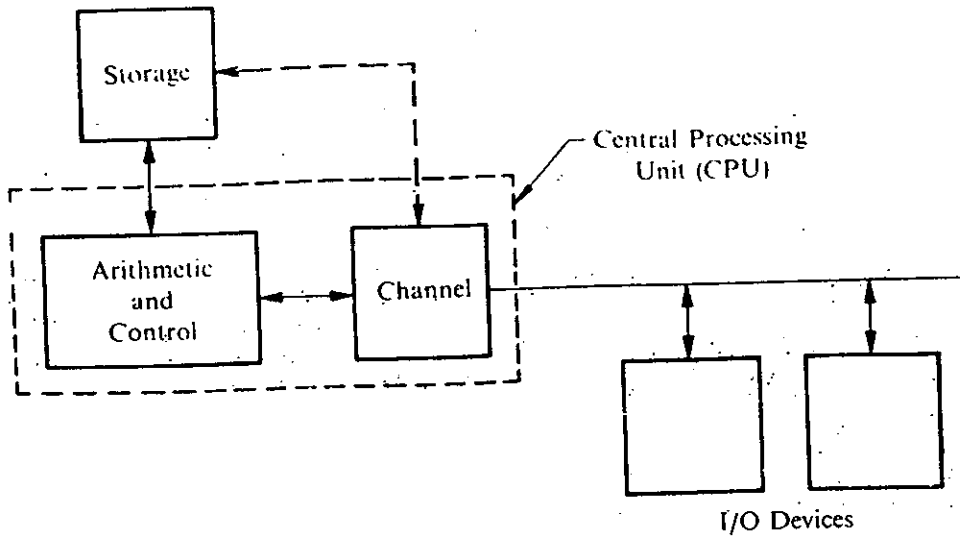

Figure #4 Key Computer System Components

9. Central Processing Unit (CPU)

- The central processing unit or CPU performs three essential jobs for the computer system. These are the ***control*** section, the ***arithmetic*** section and the ***computer control panel.***

The Control Section - controls the computer operations and interfaces.

- The control section makes all computer operation decisions by ***executing*** the programmed ***instructions*** with ***data*** read from memory or received from the input subsystem.

- The control section is a group of circuits which allows the interpretation of the defined instruction set and the development of ***resulting commands*** which are then sent to logic circuits, memory systems and input/output devices.

The Arithmetic Section - performs all arithmetic operations for the computer.

- The arithmetic section does the actual arithmetic ***problem solving*** (e.g. add or subtract) and manipulates the data as directed by the control section.

- The arithmetic section contains ***registers*** (i.e. think of the register as a dynamic data word) that hold the results of the calculations and logic elements (electronic circuitry) so that the data in the registers can be combined.

- The registers of the computer allow for the ***temporary storage*** of data, the shifting or ***arithmetic manipulation*** of that data, loading of ***instructions*** as well as ***addressing*** functions and ***transfer*** of the final data to a target destination.

Control Panel - indicates the computer ***status*** and ***diagnostics***

- The computer control panel provides the operator with ***direct access*** to the CPU ***registers*** and ***memory***. Panel switches and indicators allows register contents to be examined and/or set to check or modify ***computer status*** or the ***program progress***. Usually, the first power-up or ***restart*** for the computer would be initiated from the computer control panel.

10. Output Subsystem

- The Output Subsystem allows the computer to ***drive field devices*** or ***write to devices*** so that data generated inside the computer can be dynamically applied to the plant equipment to effect plant control functions.

- Typical output devices may be ***digital to analog*** converters (DAC's), ***Digital Outputs*** (DO's), ***Pulsed Output Points*** (POP's) graphics ***displays***, ***printers***, data links to other computers, ***storage devices*** (tape, disc, etc).

- The data output subsystem must be ***reliable*** and ***fast*** enough to initiate control parametric changes to provide adequate control response times without burdening the computer operation (i.e. loading down).

- The output system must also provide sufficient barriers to protect the computer from ***field generated faults*** and ***electrical disturbances.***

- At the same time, checks should be made to ensure that computer generated signal changes are ***not a source of electrical noise*** or fault signals for the other plant equipment.

- Precautions that can be taken to minimize such an introduction of noise would include ***signal routing***, ***device specification***, ***suppression techniques***, ***modes of operation***, etc

- Adequate ***redundancy*** of control signals should be provided to prevent a ***single failure*** from disabling the automatic computer control mode.

Digital Control Introduction Assignment

1. A digital input system is organized to read 16 digital inputs by way of a 16 bit register in the subsytem. What decimal number equivalent can be expected if all of the odd bits are set (value = 1) and all of the odd bits are off (value = 0). First sketch the register to show the binary bit pattern, then determine the hexadecimal value and finally convert that value to the decimal equivalent. Treat the odd bits as bit position 1,3,5.....15 and the even bits as bit position 0,2,4....14 with bit 0 the LSB and bit 15 the MSB.

2. A computer instruction is known to be 6643 octal. What will this instruction be in hexadecimal - sketch the 16 bit pattern.

3. What sort of performance problems could possibly be encountered if a programmer used a dynamic data area to store instructions words for future execution purposes? What would be a good general programming rule to follow?

4. Why is it important for an Input subsystem to be fast enough to representatively convert field parameter changes? How would the lack of this responsiveness effect the controlability of the loop?

5. Why is it important to take the precaution to ensure that one field input electrical problem (such as grounding) can not disrupt the entire input subsystem. Use a multiple input, computer control application as the example to explain your answer.

6. List some illustrative industrial computer input and output devices that may be encountered in an instrumentation or control application.

7. In general, how could a computer driven device introduce electrical noise into the plant? What are some precautions that could be taken to minimize this effect?

8. A control decision requires that the equivalent of 4728 decimal to be output as a hexadecimal number. Convert 4728 decimal to the equivalent hexadecimal value and then show the expected bit pattern for a 16 bit register that will output this value. (i.e. show the binary equivalent for this hexadecimal number).