Control System Training

MODULE 4 – Combinatorial Boolean Logic

Copyright Notice

These training materials, including the samples, exercises, and solutions, are copyrighted materials. Any reproduction, or use of any kind without the specific written approval of the author is strictly prohibited.

Permission for extra-curricular use by First FRC teams for FRC related training is granted, provided the original copyright and acknowledgements are retained.

© Jim Simpson, 2018

Combinatorial Boolean Logic

Definitions:

- **Boolean** – Only values are: ZERO / ONE or TRUE/FALSE

- **Combinatorial Logic** – Outcome depends only on the current value of the inputs. Nothing depends on time (or previous values of the inputs or outputs).
  - Sometimes called **Combinational Logic**
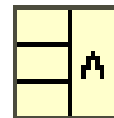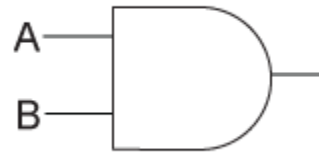
Truth Table

- **Table of all possible input values and the resulting output values.**

- **For "n" inputs, the table will have 2^n rows. (A 3 input table will have 8 rows.)**

- **A table can have multiple outputs for the same inputs if needed.**

Input 1 – A	Input 2 – B	Input 3 – C	Output 1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

AND Gate

AND GATE

Input		Output
A	B	
0	0	0
0	1	0
1	0	0
1	1	1

- **Output is true when all inputs are true.**

- **Boolean algebra written as AB or A · B or A * B**

OR Gate

OR GATE

Input		Output
A	B	
0	0	0
0	1	1
1	0	1
1	1	1
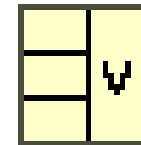
- **Output is true when any input is true**
- **Boolean algebra written as A+B**

NOT Gate

NOT GATE

INPUT	Output
0	1
1	0
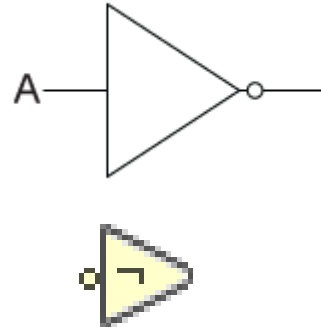
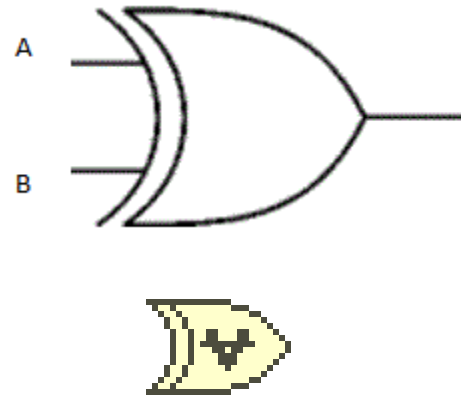
- **Output is the opposite of the input.**
- **Boolean algebra written as ^A or $\overline{A}$**

XOR (Exclusive OR) Gate

XOR GATE

Input		
A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

- **Output is true when inputs are different.**

- **Boolean algebra written as A⊕B**

- **Equivalent to:**
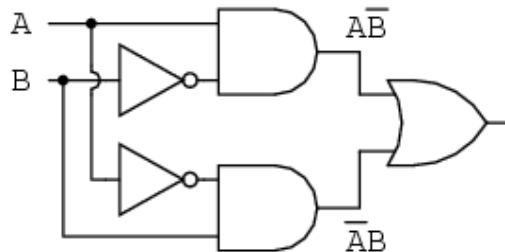
Other Gates

- **NAND – Not And**

- **NOR – Not Or**

- **Not XOR – Not Exclusive OR**

Boolean Algebra

- **Write out as regular algebra. Similar to numeric expression**

- **Symbols**
  - + means OR
  - * means AND
  - • means AND
  - Line over term or ^ means NOT
  - ⊕ means XOR (exclusive OR)
  - = means equals

- **Parenthesis can be used for grouping, similar to regular algebra**

- **Examples:**
  - OUT = PERM • INPUT
  - H = A • B + C • D • (E + F + G)

Boolean Algebra - Rules

				Law
1a.	$X \cdot 0 = 0$	1b.	$X + 1 = 1$	Annulment Law
2a.	$X \cdot 1 = X$	2b.	$X + 0 = X$	Identity Law
3a.	$X \cdot X = X$	3b.	$X + X = X$	Idempotent Law
4a.	$X \cdot \overline{X} = 0$	4b.	$X + \overline{X} = 1$	Complement Law
5.	$\overline{\overline{X}} = X$			Double Negation Law
6a.	$X \cdot Y = Y \cdot X$	6b.	$X + Y = Y + X$	Commutative Law
7a.	$X(YZ) = (XY)Z = (XZ)Y = XYZ$			Associative Law
7b.	$X + (Y + Z) = (X + Y) + Z = (X + Z) + Y = X + Y + Z$			Associative Law
8a.	$X \cdot (Y + Z) = XY + XZ$	8b.	$X + YZ = (X + Y) \cdot (X + Z)$	Distributive Law
9a.	$\overline{X \cdot Y} = \overline{X} + \overline{Y}$	9b.	$\overline{X + Y} = \overline{X} \cdot \overline{Y}$	de Morgan's Theorem
10a.	$X \cdot (X + Y) = X$	10b.	$X + XY = X$	Absorption Law
11a.	$(X + Y) \cdot (X + \overline{Y}) = X$	11b.	$XY + X\overline{Y} = X$	Redundancy Law
12a.	$(X + \overline{Y}) \cdot Y = XY$	12b.	$X\overline{Y} + Y = X + Y$	Redundancy Law
13a.	$(X + Y) \cdot (\overline{X} + Z) \cdot (Y + Z) = (X + Y) \cdot (\overline{X} + Z)$			Consensus Law
13b.	$XY + \overline{X}Z + YZ = XY + \overline{X}Z$			Consensus Law
14a.	$X \oplus Y = (X + \overline{Y}) \cdot (\overline{X} + Y)$	14b.	$X \oplus Y = \overline{X}Y + X\overline{Y}$	XOR Gate
15a.	$X \odot Y = (X + Y) \cdot (\overline{X \cdot Y})$	15b.	$X \odot Y = \overline{X}\,\overline{Y} + XY$	XNOR Gate
15c.	$X \odot Y = (X + Y) \cdot (\overline{X} + \overline{Y})$			XNOR Gate

- **There are other rules.**

Boolean Algebra – Creation from Truth Table

■ **Step 1: Create Truth Table for all possible states**

■ **Step 2: For each 1 output write the equations:**

Input 1	Input 2	Input 3	
A	B	C	Output 1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

^A * B * C (row 0 1 1)

A * ^B * C (row 1 0 1)

A * B * ^C (row 1 1 0)

A * B * C (row 1 1 1)

■ **Step 3: Write complete output equation as "sum of products"**

OUTPUT 1 = ^A * B * C + A * ^B * C + A * B * ^C + A * B * C

Boolean Algebra – Creation from Truth Table

■ **Step 4 (optional): Use boolean algebra and rules to simplify**

OUTPUT 1 = ^A * B * C + A * ^B * C + A * B * ^C + A * B * C

simplifies to

OUTPUT 1 = B * C + A * C + A * B

Boolean Algebra – Creation from Truth Table

■ **A similar process can be used to create a "product of sums". 1) Write equations for each of the 0 outputs. 2) AND these equations together.**

Input 1	Input 2	Input 3	
A	**B**	**C**	**Output 1**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(A + B + C)
(A + B + ^C)
(A + ^B + C)

(^A + B + C)

■ **This yields:**

OUTPUT 1 = (A + B + C) * (A + B + ^C) * (A + ^B + C) * (^A + B + C)

Converting Numeric To Boolean

- **The output of a comparison of two numeric values is a boolean.**

- **Examples:**

  - **Equal –** True when **A = B**

  - **Not Equal –** True when **A <> B**

  - **Greater or Equal –** True when **A >= B**

  - **Less or Equal –** True when **B <= A**

  - **In Range –** True when **C >= A and C <= B.** For the output to ever be true **B** must be **> A.**

- **Enhanced comparison functions often have a deadband value to reduce chatter and eliminate insignificant differences.**

Sample 4.1 – System Definition

- **A system is defined as follows:**

  - System has two modes of operation "manual" and "automatic". A user operated switch selects "automatic" mode.

  - An external program generates a "start command" for use when in "automatic" mode.

  - A user operated button "start command" is used when in "manual" mode.

  - A user operated "stop" switch stops the output in either mode.

  - An external sensor provides "permissive" for the system to operate.

- **Calculate the system operate (run) output from the defined inputs.**

Sample 4.1 – Create Truth Table

■ **From the system definition and operation description:**

- **List all the inputs and outputs**

- **Create a truth table (or write the equations directly if possible).**

User Stop Switch	Automatic Switch	Automatic Run Request	User Run Request	Permissive Sensor	System Run output
STOP	AUTO	A_RUN	U_RUN	PERM	RUN
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	1
0	1	1	1	0	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	0
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

Sample 4.1 – Create Sum of Products

■ **For each True output in the truth table create the equation and combine**

User Stop Switch	Automatic Switch	Automatic Run Request	User Run Request	Permissive Sensor	System Run output	
STOP	**AUTO**	**A_RUN**	**U_RUN**	**PERM**	**RUN**	
0	0	0	1	1	1	^STOP * ^AUTO * ^A_RUN * U_RUN * PERM
0	0	1	1	1	1	^STOP * ^AUTO * A_RUN * U_RUN * PERM
0	1	1	0	1	1	^STOP * AUTO * A_RUN * ^U_RUN * PERM
0	1	1	1	1	1	^STOP * AUTO * A_RUN * U_RUN * PERM

RUN = (^STOP * ^AUTO * ^A_RUN * U_RUN * PERM) + (^STOP * ^AUTO * A_RUN * U_RUN * PERM) +
(^STOP * AUTO * A_RUN * ^U_RUN * PERM) + (^STOP * AUTO * A_RUN * U_RUN * PERM)

■ **Optionally, use boolean algebra rules to simplify**

RUN = ^STOP * PERM * (^AUTO * U_RUN + AUTO * A_RUN)
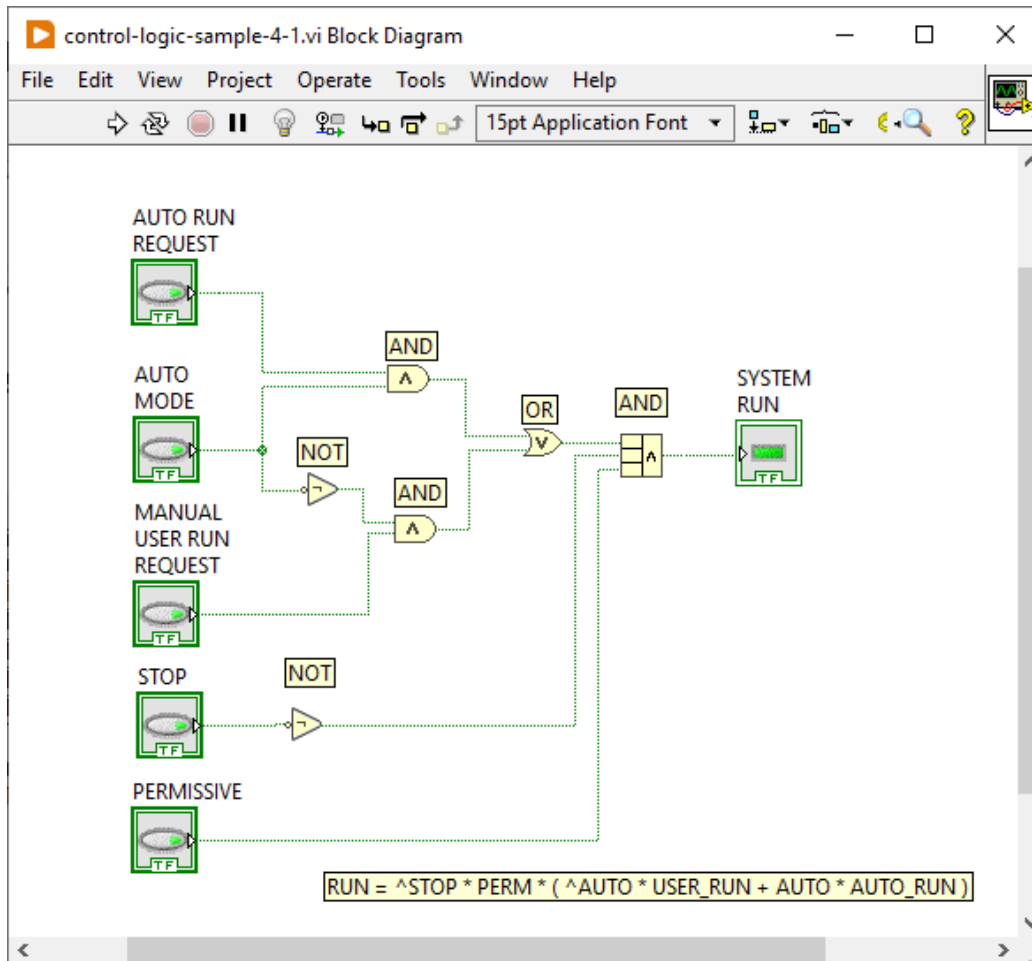
Sample 4.1 – Draw Logic Diagram

- **Draw Logic Diagram**
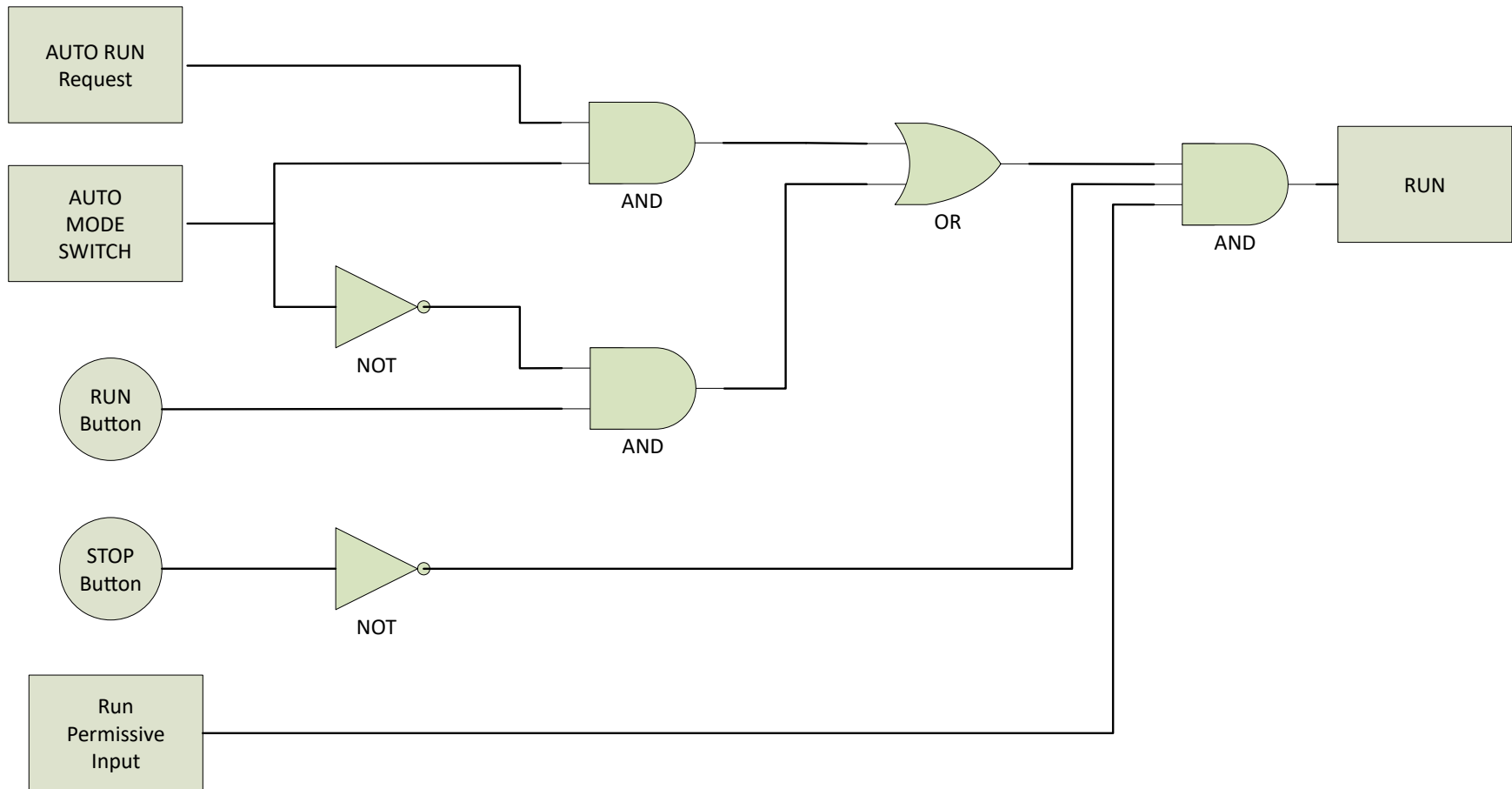
$$RUN =\ \text{^STOP} * \text{PERM} * (\ \text{^AUTO} *\ \text{U\_RUN}\ +\ \text{AUTO} * \text{A\_RUN}\ )$$

Sample 4.1 – LabVIEW Code

■ **LabVIEW Block Diagram**

Exercise 4.1 – Ball shooter size detector

- Floor contains 3 sizes of balls. Only the middle size can be shot correctly.
  - The small balls have 80% the diameter of the middle ball
  - The large balls have 120% the diameter of the middle ball.

- The robot picks up any size balls and they roll along a belt. The belt is always on. When the balls reach the "front" sensor either "reject" the ball back onto the floor, or "shoot" the ball.

- Boolean Inputs (sensors):

  1) Front of ball   2) 97% diameter of middle ball
  3) 103% diameter of middle ball   4) Robot enabled (simulate with a sensor)

- Boolean Outputs (actuators):

  1) Reject ball   2) Shoot ball

- When a ball is picked up it first encounters the 103% sensor, then the 97% sensor, then the front of ball sensor. Based on the size of the ball, multiple sensors may indicate true at the same time. When the front of ball sensor is TRUE either reject or shoot the ball based on its size. If the robot is not enabled, always reject the ball. Draw the logic and write the equation.

Robot Training 01

- **Complete Robot Training 01 presentation**

Exercise 4.2 – Ball shooter size detector

- **Implement the solution to 4.1 on a robot.**

- **The limit switch inputs use:**

    - Front of ball limit switch – DIO 0

    - 97% of ball limit switch – DIO 1

    - 103% of ball limit switch – DIO 2

    - Robot enabled limit switch – DIO 3

- **The DIO outputs (digital outputs) use:**

    - Shoot ball – DIO 8

    - Ball Detected – DIO 7

- **Use the robot project "put-name-here". The only VI that need to be modified are in the "BallDetectShoot" sub-directory. They are:**

    - BallDetectShoot_Open   - One time initialization goes here

    - BallDetectShoot_Execute   - Code to periodically execute goes here