HOMEWORK4

- LAB1: TRAFFIC LIGHT
- LAB2: TOTALIZER

จัดทำโดย

นายธนกฤต	**จาดทอง**	63070501208
นายลัทธพล	**ปิยะสุข**	63070501216
นางสาวฉันท์สินี	**เมืองหนู**	63070501221

INC251

ภาคการศึกษาที่ 2/2564

อ.สมชัย ตรีรัตนจารุ

4.1 TRAFFIC LIGHT

Name: Thanakrit Jadthong Student ID: 63070501208 Group: INC A01
Name: Latthaphon Piyasuk Student ID: 63070501216 Group: INC A01
Name: Chansinee Mueangnu Student ID: 63070501221 Group: INC A01

Homework 4

Write the program for howork4. Then, achieve the project name as: HW4_group Axx and report. Please send to email: inc251.2a@gmail.com

4.1 Traffic Light (S7 Graph)

- Write the IEC standard program according to the job descriptions as state below.

- Write a stage diagram for traffic light. Then, write the Program (S7-Graph) and HMI (WinCC) for this application. (You must create "AUTO/Manual" button and "STEP" button on WINCC screen.)

- Instructions

- A traffic light system is designed to control traffic flowing into an intersection of two roads.
- The system should perform the following actions.
  - When the switch, AUTO/MANUAL is 'on', the traffic lights follow the sequence shown in Figure 1. The activation of the pushbutton STEP has no effect on the system.
  - When the switch, AUTO/MANUAL is turned off, the current operation state of the traffic light system is held until the pushbutton STEP is pressed. Then the system enters the next operation state and so on.
  - If there is a power failure and recovery (CPU stop and switch to run), only YELLOW1 and YELLOW2 are flashing (**1 Hz**) for ten second (the other lights are off) before the system restarts from the first state of the operation sequence.

Name: Thanakrit Jadthong Student ID: 63070501208 Group: INC A01
Name: Latthaphon Piyasuk Student ID: 63070501216 Group: INC A01
Name: Chansinee Mueangnu Student ID: 63070501221 Group: INC A01

You should name your inputs and outputs according to their functions.

Try to minimize the number of timers used in your program.

RED1			GREEN1	YELLOW1	RED1
GREEN2	YELLOW2	RED2			
8 seconds	4 seconds	1 seconds	8 seconds	4 seconds	1 seconds

Figure 1: Traffic light operation sequence

Inputs	Outputs
AUTO/MANUAL: I125.0	RED1: Q 125.0
STEP: I125.1	YELLOW1: Q 125.1
	GREEN1: Q 125.2
	RED2: Q 125.3
	YELLOW2: Q 125.4
	GREEN2: Q 125.5

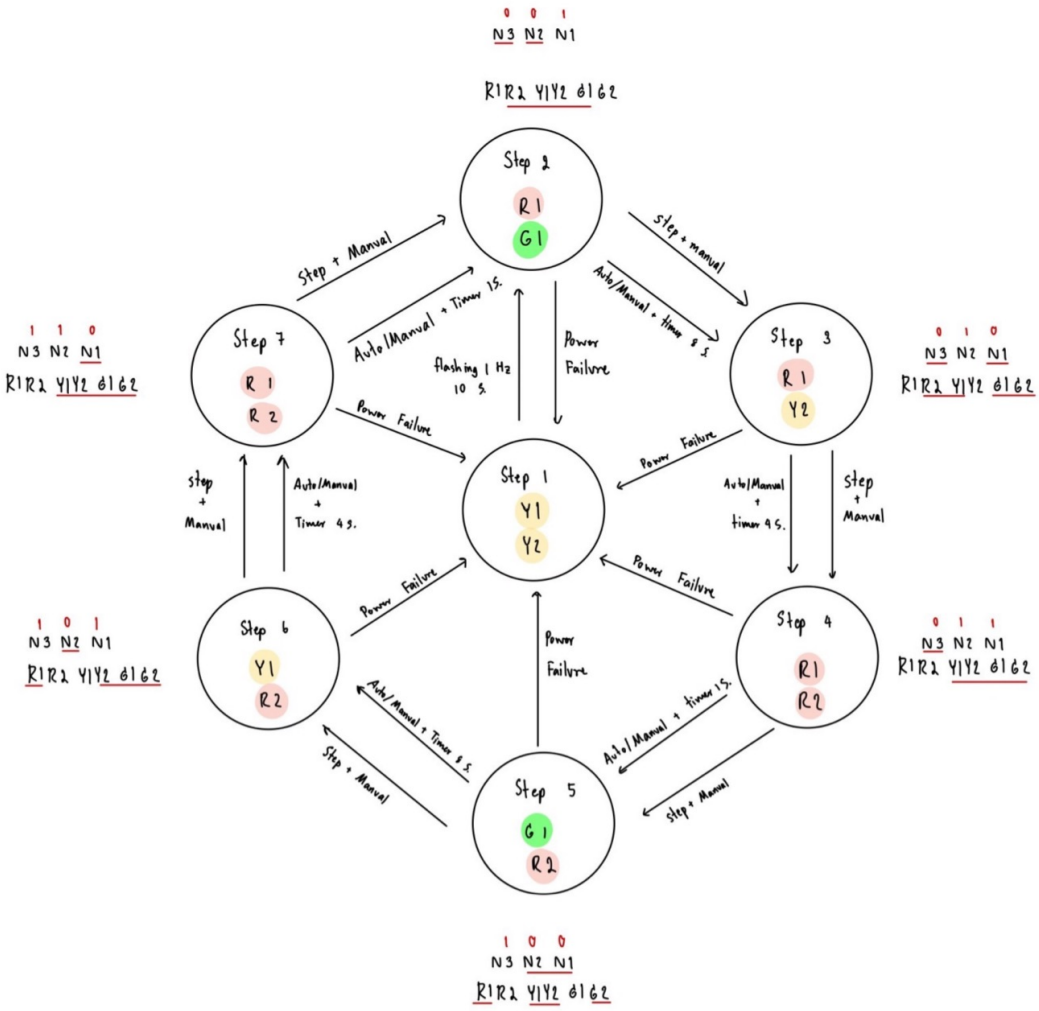
Name: Thanakrit Jadthong Student ID: 63070501208 Group: INC A01
Name: Latthaphon Piyasuk Student ID: 63070501216 Group: INC A01
Name: Chansinee Mueangnu Student ID: 63070501221 Group: INC A01

➢ Symbol:

	Statu	Symbol	Address	Data type	Comment
1		AUTO MODE	M 20.0	BOOL	
2		auto/manual	I 125.0	BOOL	
3		COMPLETE REST...	OB 100	OB 100	Complete Restart
4		G7_STD_3	FC 72	FC 72	
5		GREEN1	Q 125.2	BOOL	
6		GREEN2	Q 125.5	BOOL	
7		MANUAL MODE	M 20.1	BOOL	
8		MANUAL/AUTO	FC 1	FC 1	
9		memSTEP	M 20.3	BOOL	
1		RED1	Q 125.0	BOOL	
1		RED2	Q 125.3	BOOL	
1		reset state	M 18.0	BOOL	
1		step	I 125.1	BOOL	
1		TIME_TCK	SFC 64	SFC 64	Read the System Time
1		Y1 cond	M 17.0	BOOL	
1		Y1 cond2	M 17.2	BOOL	
1		Y2 cond	M 17.1	BOOL	
1		Y2 cond2	M 17.3	BOOL	
1		YELLOW1	Q 125.1	BOOL	
2		YELLOW2	Q 125.4	BOOL	

ตารางที่ 1: สัญลักษณ์ที่ใช้ในการเขียนโปรแกรม (S7-300) Traffic Light ทั้งหมด

➢ State Diagram:



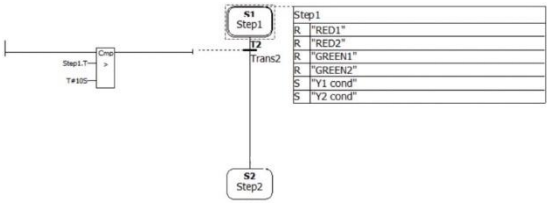
รูปที่ 1: state diagram for Traffic Light

➢ Program(LADDER & S7-Graph):

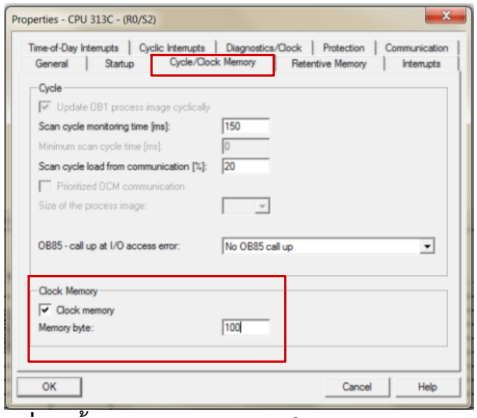
การเขียนโปรแกรมควบคุมจังหวะสัญญาณไฟจราจรนี้มีรายละเอียด ดังต่อไปนี้

- STEP1 : เป็น step ที่ไฟสีเหลือง (YELLOW1 & YELLOW2) กระพริบ 10 วินาที ด้วยความถี่ 1 hz

การเขียนโปรแกรมในขั้นตอนนี้เราจะใช้การเขียนโปรแกรมทั้ง S7-Graph และ LADDER ร่วมกัน เนื่องจากเราไม่สามารถที่จะสั่งให้ไฟสีเหลืองกระพริบได้ถ้าหากเขียนโปรแกรมแค่เพียงใน S7-Graph เพราะ S7-Graph เขียนโปรแกรมสั่ง output ได้แค่ทางเดียว (สั่งให้ output ติด/ดับ โดยใช้คำสั่ง S/R) แต่ในที่นี้เราต้องการให้ไฟสีเหลืองกระพริบ นั่นคือ จะไม่สามารถใช้คำสั่ง set ในการสั่งให้ไฟเหลืองกระพริบได้ แต่จะใช้คำสั่ง set ไปสั่งให้ Y1 cond และ Y2 cond ทำงาน โดยมีเงื่อนไขในการเปลี่ยน step คือ ถ้าหากทำงานใน STEP1 มากกว่า 10 วินาที STEP2 ถึงจะทำงานต่อ ดังรูปที่ 2 โดย Y1 cond และ Y2 cond เป็น memory ที่สร้างขึ้นมาเพื่อนำไปใช้งานในการเขียน LADDER ควบคุมให้ไฟติดกระพริบต่อไป



รูปที่ 2: เงื่อนไขสัญญาณไฟจราจร STEP1 ใน S7-Graph



รูปที่ 3: ตั้งค่า memory byte ใน clock memory
(config. Hardware: CPU → clock memory)

หลังจากที่เขียนโปรแกรม S7-Graph ให้ Y1 cond และ Y2 cond ทำงานเป็นเวลา 10 วินาทีแล้ว เราจะนำ memory ทั้งสองนี้ไปเขียนโปรแกรม LADDER เพื่อสั่งให้ไฟสีเหลืองกระพริบเป็นเวลา 10 วินาที เริ่มแรกเราจะต้องไปตั้งค่า clock memory ที่เราต้องการใช้เพื่อสั่งให้ไฟกระพริบด้วยความถี่ 1 hz ก่อน โดยสามารถตั้งค่า clock memory ได้ด้วยการไป config. CPU ในที่นี้เราได้ตั้งค่า memory byte ของ clock-memory ใน byte ที่ 100 ดังรูปที่ 3 ข้างต้น

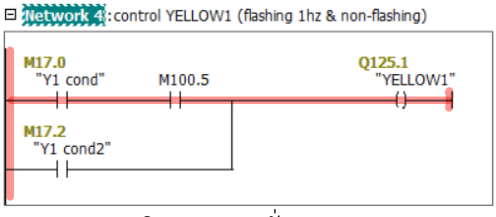
A period duration/frequency is assigned to each bit of the clock memory byte:

Bit	7	6	5	4	3	2	1	0
Period duration (s):	2	1.6	1	0.8	0.5	0.4	0.2	0.1
Frequency (Hz):	0.5	0.625	1	1.25	2	2.5	5	10

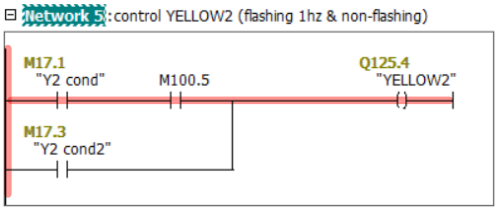
รูปที่ 4: Period Duration

(**ที่มา:** help on CPUs and Specicial FMs ในโปรแกรม S7-300)

พิจารณาจากรูปที่ 4 จะเห็นว่าถ้าหากเราต้องการสั่งให้ไฟกระพริบด้วยความถี่ 1 hz นั้น เราจะต้องใช้ bit ที่ 5 ของ memory byte ที่เราได้ตั้งค่าไว้ให้เป็น clock memory ในการสั่งงาน นั่นคือ เราจะเขียนโปรแกรม LADDER ในการสั่งให้ไฟสีเหลืองกระพริบ 1 hz โดยใช้ M100.5 ในการสั่งงาน ดังรูปที่ 5 และ 6 ซึ่งเมื่อ Y1 cond ในรูปที่ 5 และ Y2 cond ในรูปที่ 6 ทำงาน M100.5 ก็จะไปสั่งให้ไฟสีเหลืองทั้งสองดวง (YELLOW1 & YELLOW2) ติดกระพริบด้วยความถี่ 1 hz และจากที่อธิบายไปก่อนหน้านี้ว่า STEP1 ที่เขียนไว้ในโปรแกรม S7-Graph ได้สั่งให้ Y1 cond และ Y2 cond ทำงานเป็นเวลา 10 วินาที นั่นคือ ขั้นตอนต่าง ๆ ที่กล่าวไปข้างต้น จะเป็นการทำให้ไฟสีเหลืองทั้งสองดวงติดกระพริบเป็นเวลา 10 วินาที ด้วยความถี่ 1 hz



รูปที่ 5: Network4 (พิจารณาการสั่งงาน YELLOW1 is flashing)



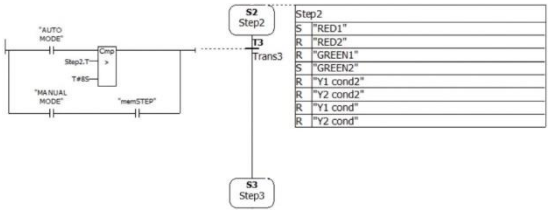
รูปที่ 6: Network5 (พิจารณาการสั่งงาน YELLOW2 is flashing)

- STEP2 : เป็น step ที่ไฟสีแดง (RED1) และไฟสีเขียว (GREEN2) ติดเป็นเวลา 8 วินาที

การเขียนโปรแกรมสั่งให้ไฟสีแดงและไฟสีเขียวติดนี้ เราได้เขียน S7-Graph และ LADDER ร่วมกัน ซึ่งในการสั่งให้ไฟสีแดงและสีเขียวติด เราจะใช้คำสั่ง set เพื่อสั่งให้ RED1 และ GREEN2 ทำงาน

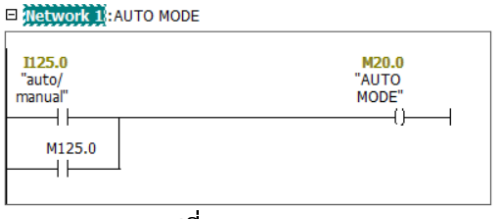
โดยใน STEP2 นี้ เราจะทำการ reset Y1 cond และ Y2 cond ที่ได้ใช้ไปใน STEP1 สำหรับสั่งให้ไฟสีเหลืองกระพริบด้วย เนื่องจาก step ที่แสดงไฟสีเหลืองกระพริบนั้นมีเฉพาะ STEP1 ที่ใช้งานแค่ครั้งเดียวในกรณีไฟตกหรือมีเหตุขัดข้องซึ่งทำให้ CPU stop แล้วเมื่อเรา reset CPU ขึ้นมาใหม่มันถึงจะเข้า STEP1 นั่นคือไม่มีการวน step ซ้ำมาที่ STEP1 ดังนั้นเราจำเป็นต้อง reset Y1 cond และ Y2 cond เพื่อที่จะทำให้ step อื่น ๆ หลังจากนี้ที่มีการสั่งงานให้ไฟสีเหลือง (YELLOW1 & YELLOW2) ติดนั้น ทำงานได้ปกติ

ส่วนเงื่อนไขเวลาที่ STEP2 ทำงานนั้น คือ 8 วินาที กรณีที่อยู่ในโหมด AUTO โดยเมื่อผ่าน 8 วินาทีนี้ไป โปรแกรมก็จะรันใน STEP3 ต่อ แต่ถ้าหากอยู่ในโหมด MANUAL การที่จะรันใน step ต่อไป จะต้องได้รับค่าจากการกดปุ่ม STEP มาก่อน ซึ่งเราได้เขียนโปรแกรม S7-Graph เพื่อควบคุมสัญญาณไฟจราจรใน STEP2 นี้ ไว้ดังรูปที่ 7

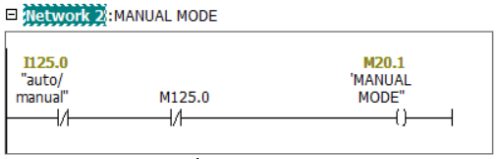


รูปที่ 7: เงื่อนไขสัญญาณไฟจราจร STEP2 ใน S7-Graph
(RED1 & GREEN2 turn on 8s)

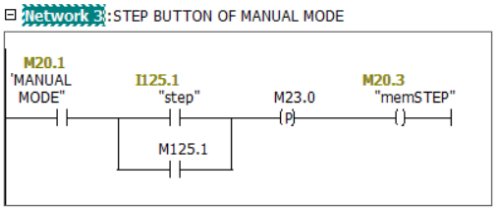
การเขียนโปรแกรมใน STEP2-STEP7 ส่วนของการกดปุ่มเพื่อเลือกโหมดระหว่าง AUTO/MANUAL และการกดปุ่ม STEP เมื่ออยู่ในโหมด MANUAL เพื่อเปลี่ยน step การทำงานนั้น เราจะเขียนโปรแกรมไว้ใน LADDER เป็นหลัก และเก็บค่าที่รับมาจากการกดปุ่มไว้ใน memory ซึ่งจะเรียกไปใช้งานในการเขียน S7-Graph เพื่อควบคุมสัญญาณไฟจราจรต่อไป โดยการเขียนโปรแกรม LADDER ควบคุมเงื่อนไขของทั้ง AUTO mode , MANUAL mode และการกดปุ่ม STEP เป็นไปตามรูปที่ 8-10 ตามลำดับ



รูปที่ 8: Network1



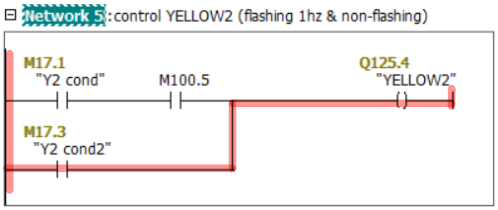
รูปที่ 9: Network2



รูปที่ 10: Network3

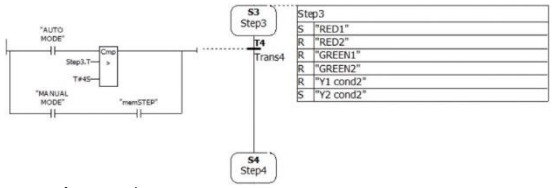
- STEP3 : เป็น step ที่ไฟสีแดง (RED1) และไฟสีเหลือง (YELLOW2) ติดเป็นเวลา 4 วินาที

การเขียนโปรแกรมสั่งให้ไฟสีแดงและไฟสีเหลืองติดนี้ เราได้เขียน S7-Graph และ LADDER ร่วมกัน โดยในการสั่งให้ไฟสีแดงติด เราจะใช้คำสั่ง set เพื่อสั่งให้ RED1 ทำงาน แต่ในการสั่งให้ไฟสีเหลืองติด (แบบไม่กระพริบ) จะไม่สามารถใช้คำสั่ง set เพื่อสั่งให้ Y1 cond & Y2 cond ทำงานได้ เนื่องจาก memory ทั้งสองนี้เราได้ใช้เพื่อสั่งงานสำหรับทำให้ไฟสีเหลืองติดกระพริบไปแล้ว เราจึงต้องสร้าง memory ขึ้นมาอีก 2 ตัว คือ Y1 cond2 และ Y2 cond2 สำหรับนำไปใช้สั่งงานต่อใน LADDER เพื่อให้ไฟสีเหลืองติดแบบไม่กระพริบ โดยโปรแกรม LADDER ที่เขียนเพื่อสั่งให้ไฟสีเหลือง (YELLOW2) ติด ใน STEP3 นี้ มาจากการที่ Y2 cond2 ทำงาน ซึ่งเขียน LADDER ได้ดังรูปที่ 11



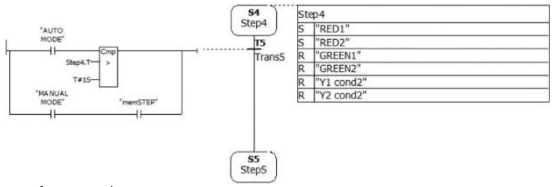
รูปที่ 11: Network5 (พิจารณาการสั่งงาน YELLOW2 is non-flashing)

ส่วนเงื่อนไขเวลาที่ STEP3 ทำงานนั้น คือ 4 วินาที กรณีที่อยู่ในโหมด AUTO โดยเมื่อผ่าน 4 วินาทีนี้ไป โปรแกรมก็จะรันใน STEP4 ต่อ แต่ถ้าหากอยู่ในโหมด MANUAL การที่จะรันใน step ต่อไป จะต้องได้รับค่าจากการกดปุ่ม STEP มาก่อน ซึ่งเราได้เขียนโปรแกรม S7-Graph เพื่อควบคุมสัญญาณไฟจราจรใน STEP3 นี้ ไว้ดังรูปที่ 12

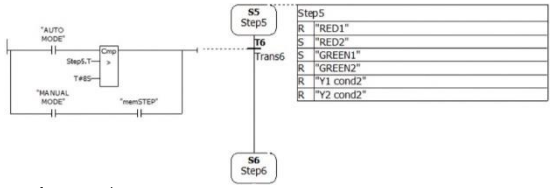


รูปที่ 12: เงื่อนไขสัญญาณไฟจราจร 3 ใน S7-Graph
(RED1 & YELLOW2 turn on 4s)

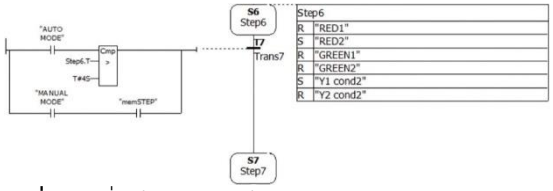
- STEP4-7 : เราได้ใช้หลักการเขียนโปรแกรมทั้งหมดเหมือนกับ STEP1-3 ดังที่กล่าวไปข้างต้น ซึ่งเป็นการเขียนโปรแกรมโดยใช้ S7-Graph (สำหรับควบคุมการแสดงสีไฟจราจรในแต่ละ step, เงื่อนไขเวลา และเงื่อนไขการเปลี่ยน step) ร่วมกับการเขียน LADDER (สำหรับควบคุมไฟสีเหลืองแบบกระพริบ/ไม่กระพริบ และการกดปุ่ม) ซึ่งโปรแกรมที่เราเขียนโดยใช้ S7-Graph ใน STEP4-7 เป็นไปดังรูปที่ 13-16 ตามลำดับ



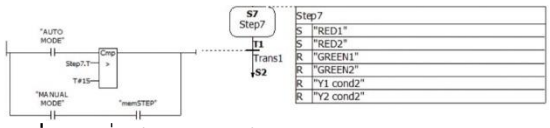
รูปที่ 13: เงื่อนไขสัญญาณไฟจราจร step4 ใน S7-Graph
(RED1 & RED2 turn on 1s)



รูปที่ 14: เงื่อนไขสัญญาณไฟจราจร step5 ใน S7-Graph
(GREEN1 & RED2 turn on 8s)

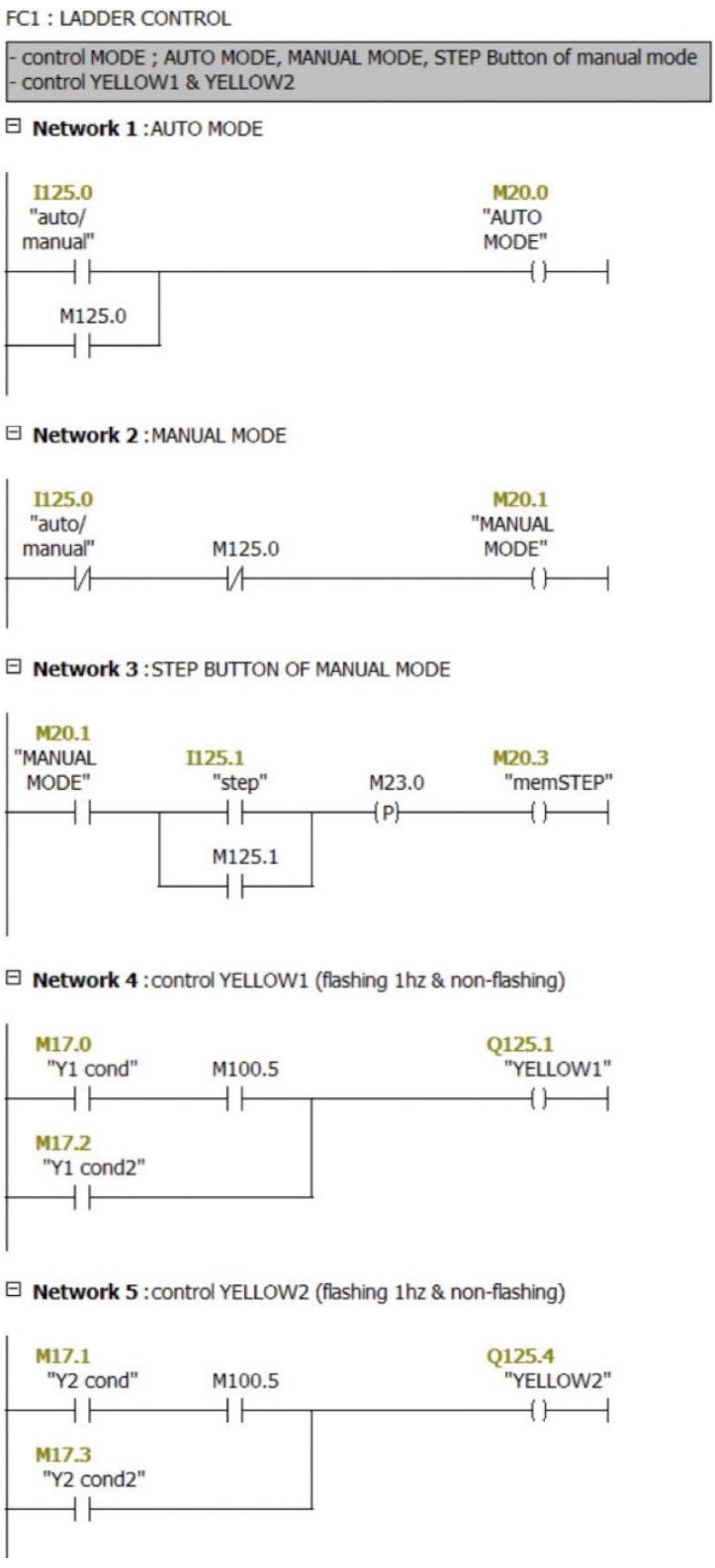


รูปที่ 15: เงื่อนไขสัญญาณไฟจราจร step6 ใน S7-Graph
(YELLOW1 & RED2 turn on 4s)



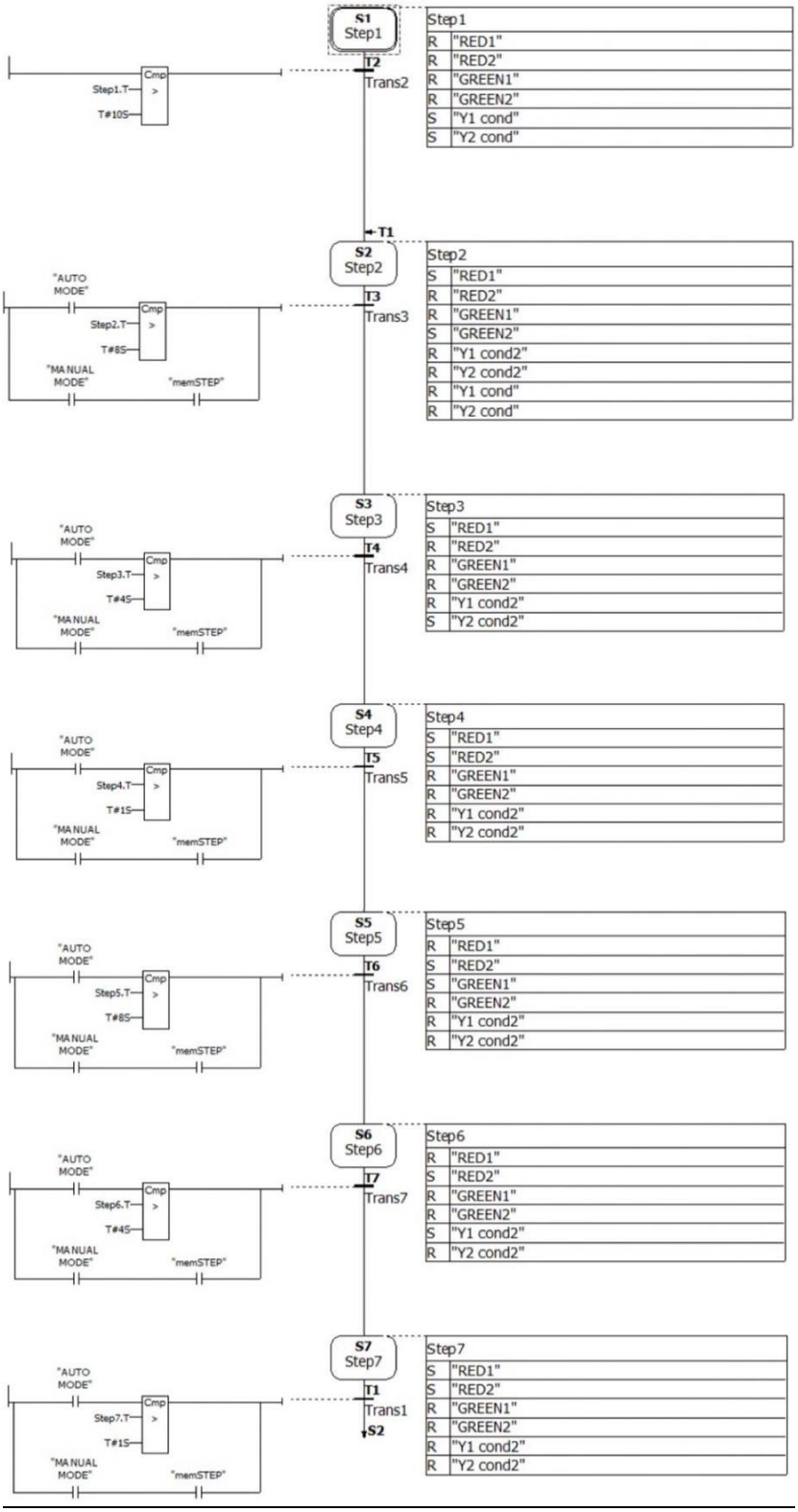
รูปที่ 16: เงื่อนไขสัญญาณไฟจราจร step7 ใน S7-Graph
(RED1 & RED2 turn on 1s)

- การเขียนโปรแกรม LADDER เพื่อควบคุมระบบ Traffic Light เราได้เขียนไว้ใน FC1 โดยโปรแกรมทั้งหมดเป็นไปดังรูปที่ 17



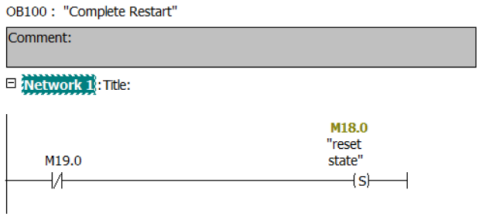
รูปที่ 17: โปรแกรม LADDER ทั้งหมดที่ใช้ในการควบคุมระบบ Traffic Light

- การเขียนโปรแกรม S7-Graph เพื่อควบคุมระบบ Traffic Light เราได้เขียนไว้ใน FB1 โดยโปรแกรมทั้งหมดเป็นไปดังรูปที่ 18



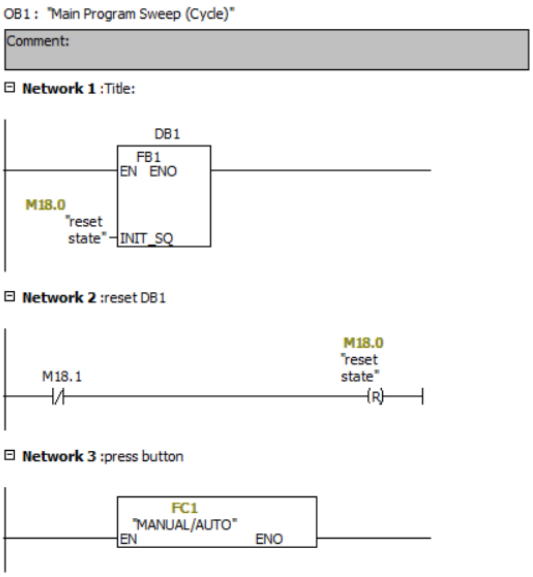
รูปที่ 18: โปรแกรมควบคุมระบบ Traffic Light ทั้งหมดที่เขียนใน S7-Graph

- เนื่องจากเราต้องการให้ เมื่อไฟตก หรือมีเหตุขัดข้องที่ทำให้ CPU stop เมื่อเรา reset CPU ขึ้นมาใหม่ มันจะเข้าสู่โหมดไฟเหลืองกระพริบ 10 วินาที (STEP1) โดยกระพริบด้วยความถี่ 1 Hz ก่อนที่จะรันโปรแกรมใน step ต่าง ๆ ต่อไป ซึ่งการที่โปรแกรมจะอยู่ใน step ใดก็ตาม แล้วเมื่อ reset CPU ขึ้นมาแล้วโปรแกรมกลับมาที่ step เริ่มต้นได้นั้น เนื่องจากเรามีการใช้ OB100 ดังรูปที่ 19



รูปที่ 19: OB100

- โดยการทำงานทั้งหมดของโปรแกรมควบคุมระบบ Traffic Light นี้ เราได้เขียนไว้ใน FC1 และ FB1 (ต้องใช้งานควบคู่กับ DB1) โดยเราได้เรียกใช้งาน FC1 และ FB1 ที่สร้างขึ้นนี้ ผ่าน OB1 ดังรูปที่ 20



รูปที่ 20: OB1

➢ HMI (WinCC):

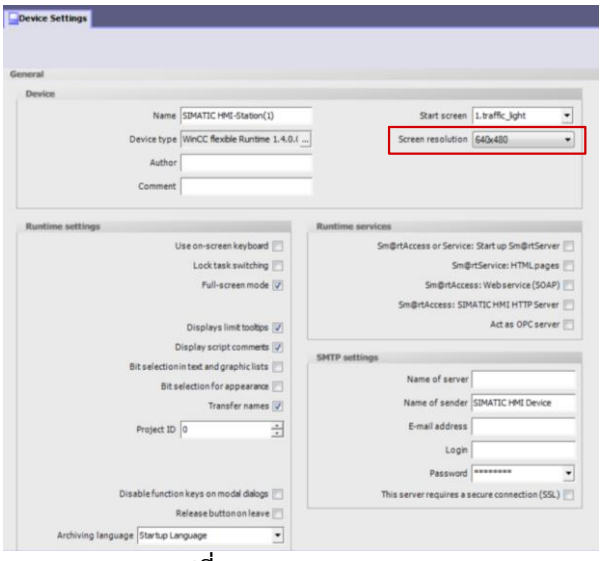
1. Tags:

Name	Display name	Connection	Data type	Symbol	Address	Array elem...	Acquisition cycle
step		Connection_1	Bool	<Undefined>	M 125.1	1	100 ms
GREEN1		Connection_1	Bool	<Undefined>	Q 125.2	1	100 ms
RED2		Connection_1	Bool	<Undefined>	Q 125.3	1	100 ms
YELLOW1		Connection_1	Bool	<Undefined>	Q 125.1	1	100 ms
AUTO/MANUAL		Connection_1	Bool	<Undefined>	M 125.0	1	100 ms
YELLOW2		Connection_1	Bool	<Undefined>	Q 125.4	1	100 ms
GREEN2		Connection_1	Bool	<Undefined>	Q 125.5	1	100 ms
RED1		Connection_1	Bool	<Undefined>	Q 125.0	1	100 ms

ตารางที่ 2: Tags ทั้งหมดที่ใช้ในการเขียนโปรแกรม (HMI) Traffic Light

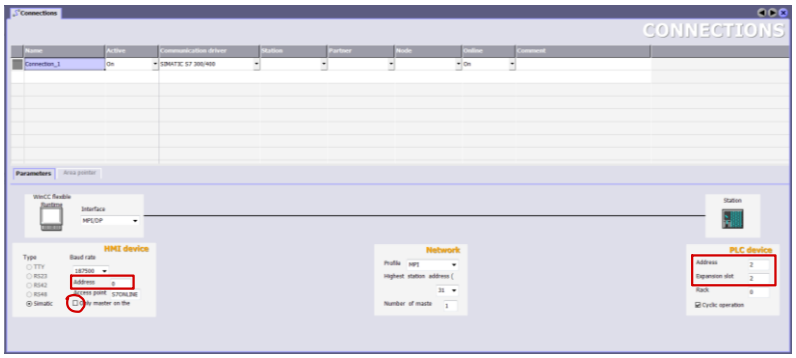
2. Setting:

- ตั้งค่า resolution ของหน้าจอ



รูปที่ 21: Device Settings

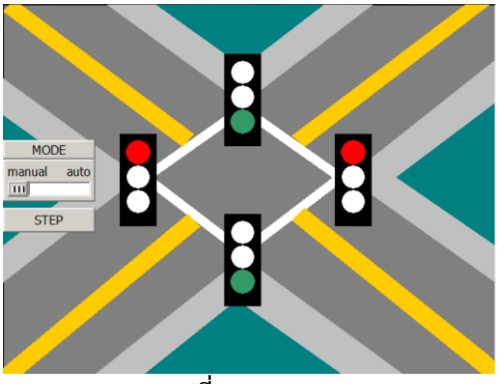
- เลือก protocol ในการสื่อสาร



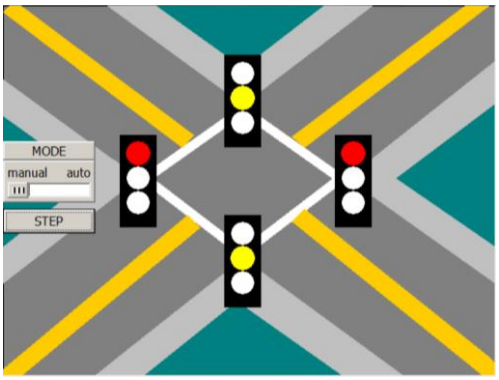
รูปที่ 22: Communication → Connections

3. HMI Screens:

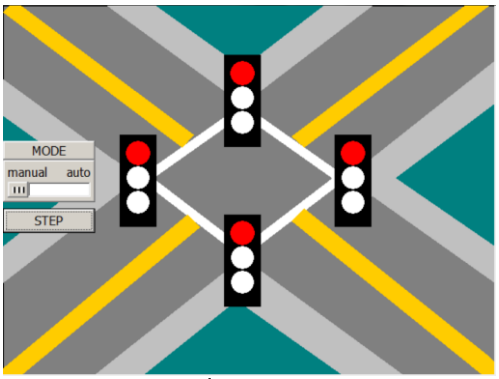
- จังหวะสัญญาณไฟจราจรในแต่ละ STEP เป็นไปตามโปรแกรมที่วางไว้ตามลำดับ ดังนี้



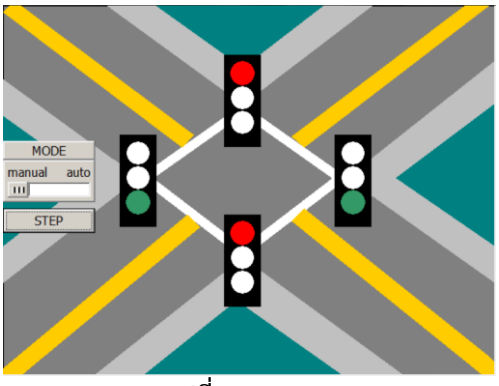
รูปที่ 23: STEP2



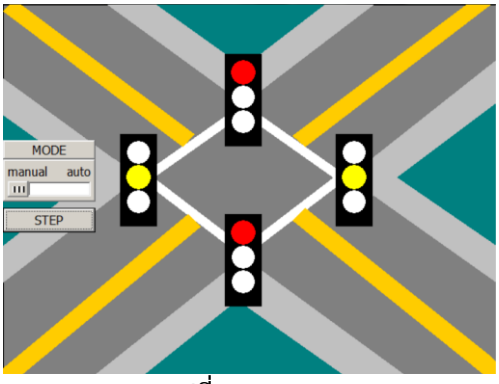
รูปที่ 24: STEP3



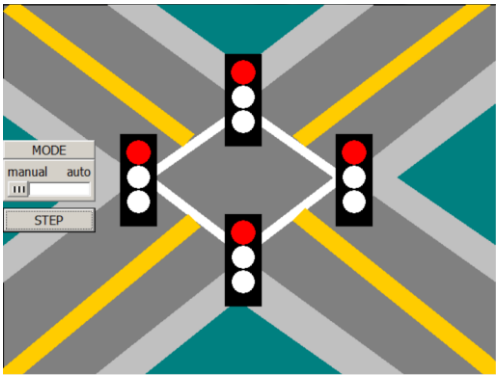
รูปที่ 25: STEP4



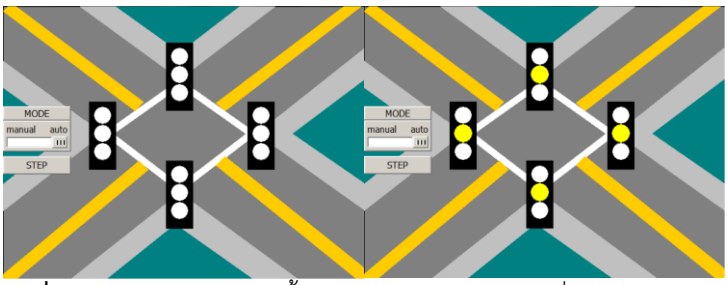
รูปที่ 26: STEP5



รูปที่ 27: STEP6



รูปที่ 28: STEP7



รูปที่ 29: STEP1 ไฟสีเหลืองทั้งหมดกระพริบ 10 วินาที เมื่อครบ 10 วินาที
จะกลับเข้าสู่ STEP2 ต่อ (STEP1 เกิดเมื่อ CPU stop แล้ว reset CPU ใหม่)

- **Questions:**

1. What is the minimum number of step required by the problem?
   ANS: 7 steps
2. What is the minimum number of timers required by the problem?
   ANS: 0 timer

- **Summary:**

โปรแกรมควบคุมระบบ Traffic Light นี้ เราได้ออกแบบโปรแกรมโดยสร้างให้มีปุ่มสำหรับเลือกโหมดระหว่าง AUTO/MANUAL ซึ่งถ้าหากเลือกโหมด AUTO การควบคุมจังหวะสัญญาณไฟจราจรจะเป็นไปตามโปรแกรมที่วางไว้โดยอัตโนมัติ โดยโปรแกรมจะรันแต่ละ step ตามเวลาที่ตั้งไว้ แต่ถ้าหากเลือกโหมด MANUAL เราจะต้องควบคุมจังหวะสัญญาณไฟจราจรด้วยตนเอง นั่นคือ จะต้องกดปุ่ม STEP เพื่อเปลี่ยน step ซึ่งโหมด MANUAL นี้ จะไม่มีการกำหนดเวลาสำหรับการเปลี่ยน step การที่จะเปลี่ยน step ได้นั้นขึ้นกับการกดปุ่ม STEP อีกทั้ง เมื่อทำการเปลี่ยนโหมด สัญญาณไฟจราจรจะทำการรัน state ต่อไปโดยอัตโนมัติ เช่น เมื่ออยู่โหมด MANUAL ใน STEP5 แล้วกดเปลี่ยนโหมดทันทีเป็นโหมด AUTO โปรแกรมจะทำการรันต่อโหมด AUTO ใน STEP6 เมื่อครบเวลาตามที่ตั้งไว้จาก STEP6 ก็จะเปลี่ยนเป็น STEP7 เมื่อครบเวลาใน STEP7 ก็จะกลับไปสู่ STEP2 และวน state ซ้ำไปเรื่อย ๆ แต่ถ้าหากไฟตก หรือมีเหตุขัดข้องที่ทำให้ CPU stop เมื่อเรา reset CPU ขึ้นมาใหม่ มันจะเข้าสู่โหมดไฟเหลืองกระพริบ 10 วินาที (STEP1) โดยกระพริบด้วยความถี่ 1 Hz ก่อนที่จะรันโปรแกรมใน step ต่าง ๆ ต่อไป

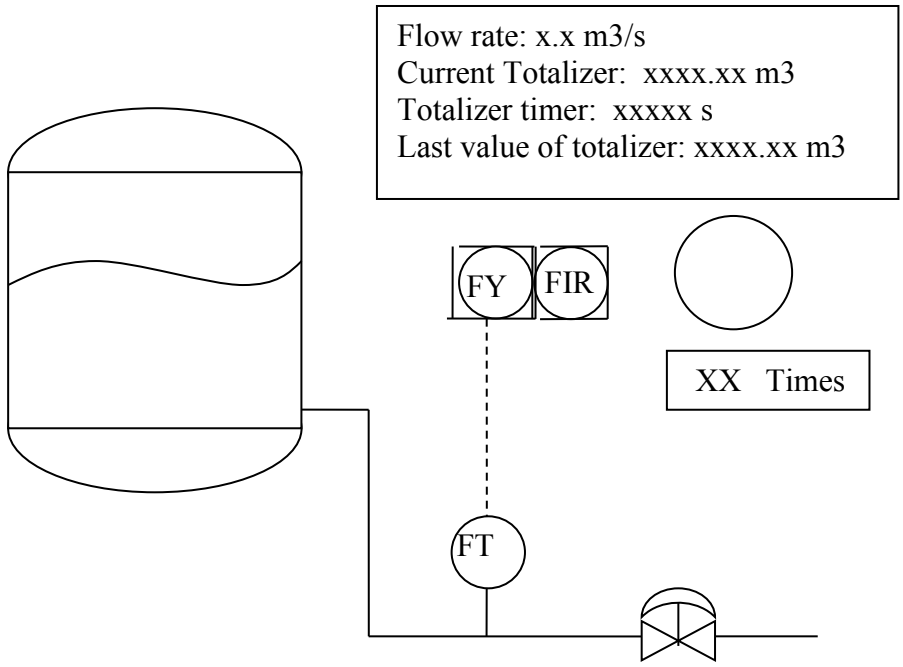
ในการเขียนโปรแกรมควบคุมระบบ Traffic Light ดังที่กล่าวไปข้างต้นนั้น เราใช้โปรแกรม S7-300 โดยการเขียน LADDER ร่วมกับการเขียน S7-Graph และมีการเขียน HMI เพื่อแสดงการทำงานของโปรแกรมควบคุมจังหวะสัญญาณไฟจราจรดังที่อธิบายไปแล้วก่อนหน้านี้

4.2 TOTALIZER

4.2 Totalizer (SCL)

Write the Program (SCL and Ladders) and HMI (WinCC) for this application. You must create "Reset" button (I125.7) on WINCC screen.

Problem



Buffer tanks are used to storage intermediate products or finished products of process control system. It is important to know the quantity of transferred product from the buffer tanks. The totalizer (FY) is used to calculate the volume of transferred fluid. The basic principle is to integrating the very accurate flow of the transferred fluid (FT).

Lab instructions

Let's using PLC as totalizer, write PLC program, using the analog signal (4-20mA) as flow signal rang 0-1.1 m^3/s. The totalizing value will be reset to zero for every 24 hrs. or by operator with reset command by repeat pressed and released the reset button (I 125.7) more than 20 times for 10 seconds.

➢ Symbol:

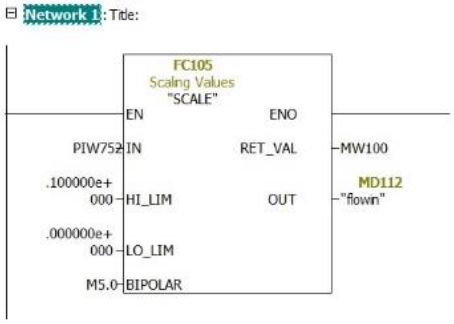
	Statu	Symbol	Address	Data type	Comment
1		AnalogSignal	FC 3	FC 3	
2		counter	MW 20	INT	
3		counterWord	MW 22	WORD	
4		crrtWater	MD 120	REAL	
5		CYC_INT5	OB 35	OB 35	Cyclic Interrupt 5
6		dummy1	M 14.0	BOOL	
7		flowin	MD 112	REAL	
8		flowout	MD 116	REAL	
9		pressedReset	M 12.0	BOOL	
1		Rcounter	M 15.0	BOOL	
1		resetButt	I 124.0	BOOL	
1		SCALE	FC 105	FC 105	Scaling Values
1		timeNow	MD 60	DINT	
1		totalNow	MD 124	REAL	
1		totalOld	MD 128	REAL	
1		VAT_1	VAT 1		
1		waterNow	FC 5	FC 5	
1					

ตารางที่ 3 : สัญลักษณ์ที่ใช้ในการเขียนโปรแกรม (s7-300) Totalizer (SCL) ทั้งหมด

➢ Program (LADDER):

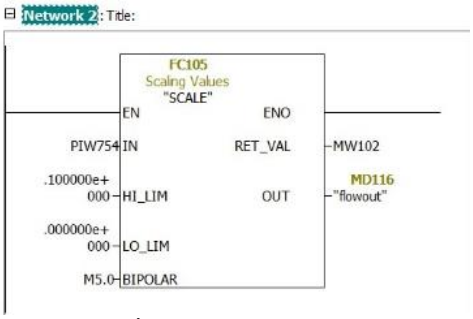
- **FC3:**

1. NETWORK1: แปลงอัตราการไหลเข้าของของเหลวจากจำนวนเต็มให้กลายเป็นจำนวนจริงโดยอยู่ในขอบเขตระหว่าง HI_LIM และ LOW_LIM ซึ่งก็คือค่าสูงสุดต่ำสุด โดยเก็บที่ MD 112



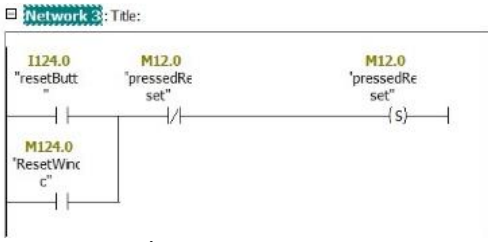
รูปที่ 30: FC3 NETWORK1

2. NETWORK2: แปลงอัตราการไหลออกของของเหลวจากจำนวนเต็มให้กลายเป็นจำนวนจริงโดยอยู่ในขอบเขตระหว่าง HI_LIM และ LOW_LIM ซึ่งก็คือค่าสูงสุดต่ำสุด โดยเก็บที่ MD 116



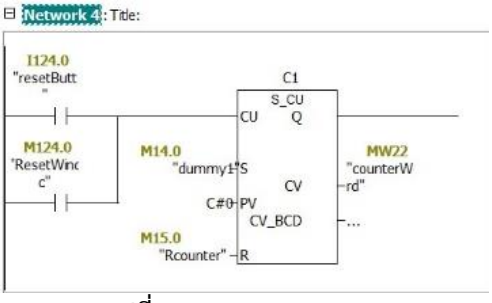
รูปที่ 31: FC3 NETWORK2

3. NETWORK3: การทำงานของปุ่ม Reset ที่กดจาก PLC(I124.0) และจาก WinCC(M124.0) ซึ่งเมื่อปุ่มถูกกดแล้วจะสั่งให้หน้า contact M12.0 ทำงาน และส่งผลให้ Network 5, 6, 7 และ 8 ทำงาน



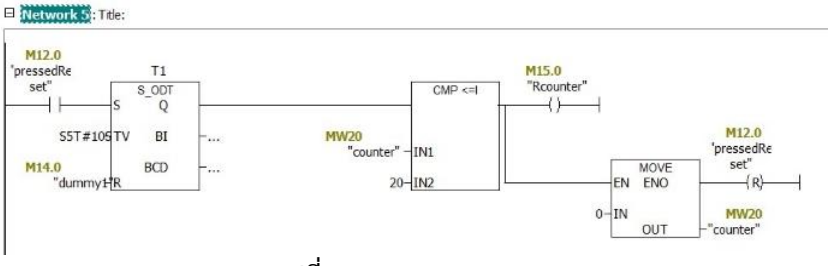
รูปที่ 32 FC3 NETWORK3

4. NETWORK4: Counter นับจำนวณครั้งที่มีการกดปุ่ม Reset ทั้งจากที่กดจาก PLC(I124.0) และกดจาก WinCC(M124.0) โดยจะบวกค่า counter ไป 1 เมื่อปุ่มถูกกดและเก็บค่า counter ที่ MW22



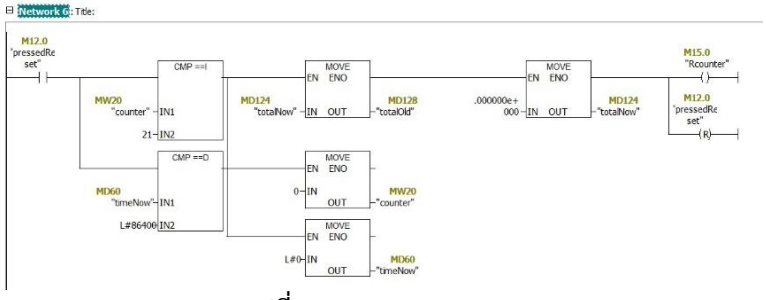
รูปที่ 33: FC3 NETWORK4

5. NETWORK5: เมื่อหน้า contact M12.0 ทำงานจะสั่งให้ ODT เริ่มนับเวลา 10 วินาทีเมื่อครบแล้วจึงสั่งให้ reset ค่า counter ผ่านหน้า contact M15.0 พร้อมทั้งส่งค่า 0 ให้กับค่า counter ที่เป็นตัวแปรชนิด Integer จากนั้นสั่งให้ reset หน้า contact M12.0 เพื่อสั่งให้หยุดการทำงานในส่วนของการทำงานของ Network 3, 5, 6, 7 และ 8



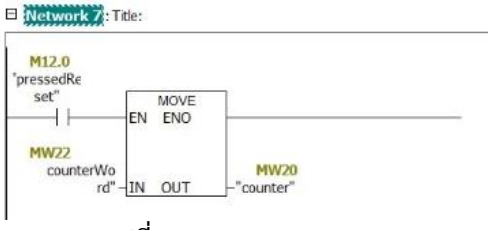
รูปที่ 34: FC3 NETWORK5

6. NETWORK6: เมื่อหน้า contact M12.0 ทำงานจะนำค่า counter และเวลาที่นับมาเปรียบเทียบ หากเงื่อนไขในข้อใดเป็นจริง (กดปุ่ม reset 21 ครั้งหรือเวลานับครบ 86400 วินาที) จะสั่งให้ส่งค่าปริมาณของเหลวในถังไปที่ปริมาณของเหลวในถังก่อนที่จะ reset ซึ่งเก็บไว้ที่คนละหน่วยความจำ จากนั้นจึงกำหนดให้ระดับของเหลวปัจจุบันมีค่าเป็น 0 m^3/s จากนั้นจึงสั่งให้ reset ค่า counter ผ่านหน้า contact M15.0 และสั่งให้ reset หน้า contact M12.0 เพื่อสั่งให้หยุดการทำงานในส่วนของการทำงานของ Network 3, 5, 6, 7 และ 8 ซึ่งในการทำงานส่วนนี้จะมีคำสั่งส่งค่า 0 ให้กับค่า counter ที่เป็นตัวแปรชนิด Integer และกำหนดค่าเวลาให้เป็น 0 วินาทีทำงานพร้อมกันไปด้วย



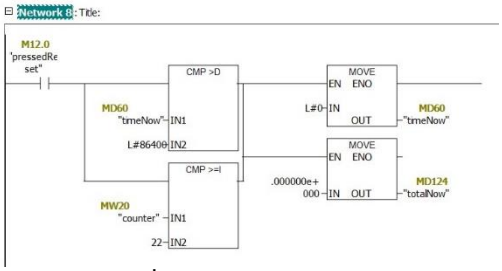
รูปที่ 35: FC3 NETWORK6

7. NETWORK7: เมื่อหน้า contact M12.0 ทำงานจะสั่งให้แปลงค่า counter ที่เป็นตัวแปรชนิด Word เป็นชนิด Integer โดยเก็บไว้ที่ MW20



รูปที่ 36: FC3 NETWORK7

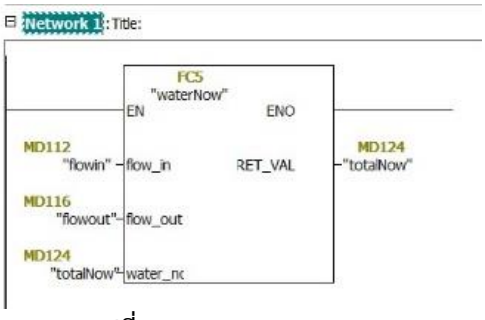
8. NETWORK8: เมื่อหน้า contact M12.0 ทำงานจะนำค่า counter และเวลาที่นับมาเปรียบเทียบ หากเงื่อนไขในข้อใดเป็นจริง (กดปุ่ม reset 22 ครั้งหรือเวลานับได้มากกว่า 86400 วินาที) จึงจะสั่งให้กำหนดค่าเวลาให้เป็น 0 วินาทีและสั่งให้ปริมาณของเหลวในถังมีค่าเป็น 0 m^3/s



รูปที่ 37: FC3 NETWORK8

- **OB35:**

1. NETWORK1: นำ Code จาก SCL มาทำงานโดยใส่ตัวแปรเก็บข้อมูลตามเงื่อนไขของโปรแกรม



รูปที่ 38: OB35 NETWORK1

2. NETWORK2: เพิ่มเวลาที่นับไป 1 โดย OB35 นี้จะทำงานทุกๆ 1 วินาทีจึงทำให้เราสามารถเพิ่มค่าเวลาได้ 1 ครั้ง ต่อ 1 วินาที

รูปที่ 39: OB35 NETWORK2

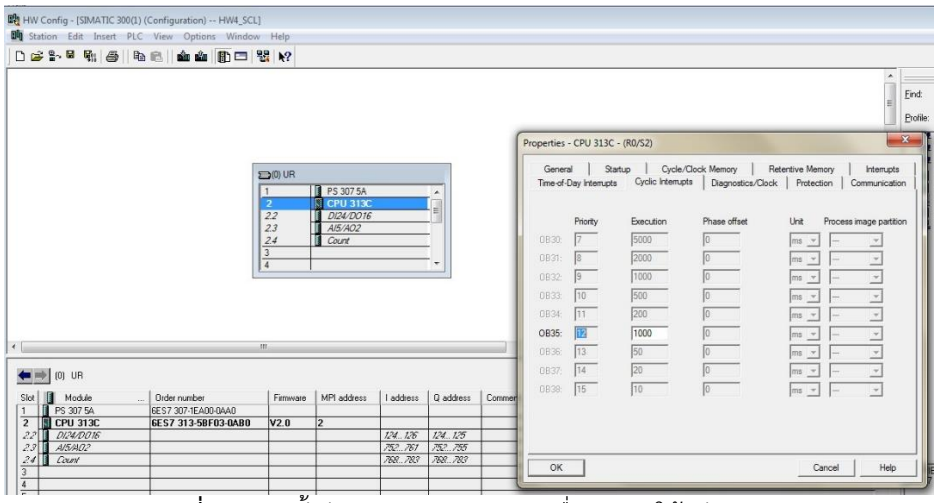
➢ <u>Program (SCL code):</u>

ในส่วนแรกกำหนดตัวแปร Input ที่จะนำมาใช้คำนวณคือ อัตราการไหลเข้าถังของของเหลว (flow in), อัตราการไหลออกจากถังของของเหลว (flow out) และปริมาณของของเหลวในถัง ณ ปัจจุบัน (water now) จากในส่วนของการทำงานจะนำปริมาณของของเหลวในถัง ณ ปัจจุบันมาบวกกับอัตราการไหลเข้าถังของของเหลวและลบออกด้วยอัตราการไหลออกจากถังของของเหลว และส่งค่าที่คำนวณได้ให้กับโปรแกรมที่เรียกใช้งาน

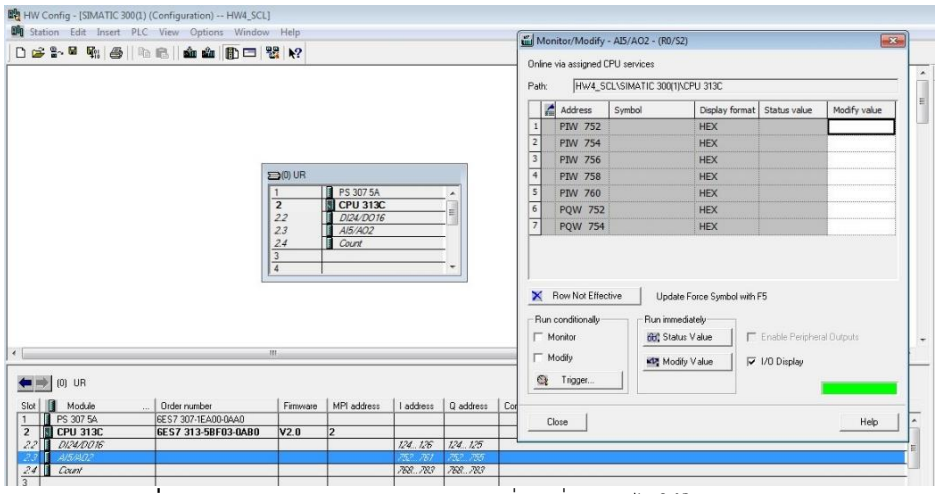
```
    FUNCTION waterNow: REAL
VAR_INPUT
    flow_in: REAL ;
    flow_out: REAL ;
    water_now: REAL;
    END_VAR
BEGIN

    waterNow:= water_now + flow_in - flow_out ;

END_FUNCTION
```

รูปที่ 40: SCL code

➢ การตั้งค่าเพิ่มเติม:



รูปที่ 41: การตั้งค่า CyClic Interrupts เพื่อกำหนดให้แต่ละรอบ
ที่ทำการ Interrupts ให้เป็น 1000 ms หรือ 1 วินาที



รูปที่ 42: แสดง Pin Analog I/O ของเครื่องเพื่อนำค่าไปใช้ในการคำนวณ

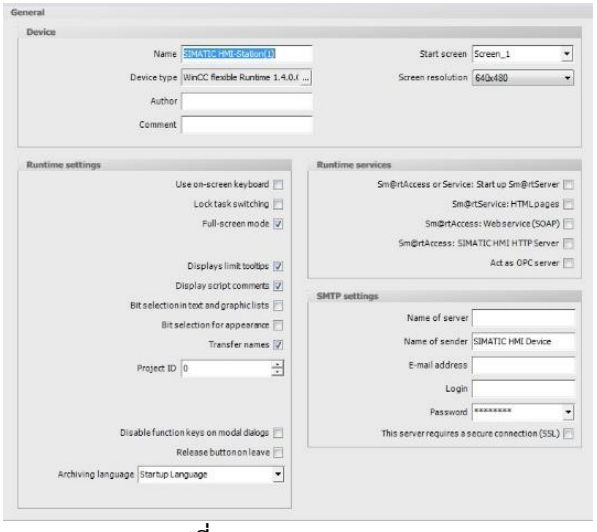
➢ HMI (WinCC):

1. Tags:

Name	Display name	Connection	Data type	Symbol	Address	Array elements	Acquisition cycle
totalOld		Connection_1	Real	<Undefined>	MD 128	1	1 s
totalNow		Connection_1	Real	<Undefined>	MD 124	1	1 s
timeNow		Connection_1	DInt	<Undefined>	MD 60	1	1 s
ResetButt		Connection_1	Bool	<Undefined>	M 124.0	1	1 s
Reset		Connection_1	Bool	<Undefined>	I 124.0	1	1 s
flowout		Connection_1	Real	<Undefined>	MD 116	1	1 s
flowin		Connection_1	Real	<Undefined>	MD 112	1	1 s
counter		Connection_1	Int	<Undefined>	MW 20	1	1 s

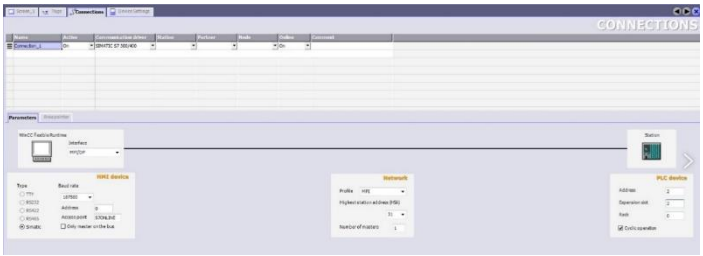
ตารางที่ 4: Tags ทั้งหมดที่ใช้ในการเขียนโปรแกรม (HMI) Totalizer

2. Setting:
   - ตั้งค่า resolution ของหน้าจอ



รูปที่ 43: Device Settings

   - เลือก protocol ในการสื่อสาร

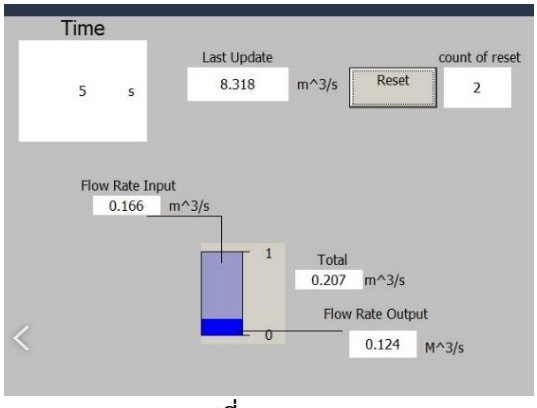


รูปที่ 44: Communication → Connections

3. HMI Screen :

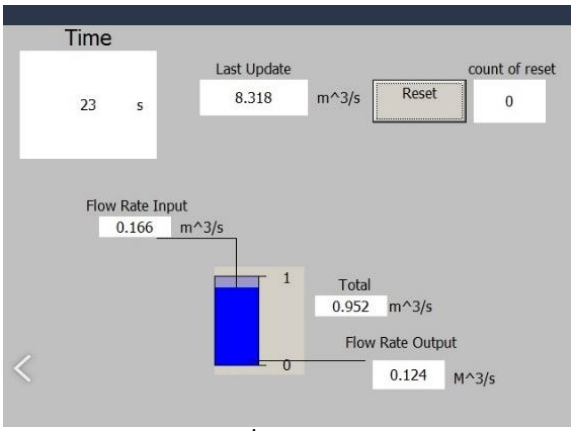
ในหน้าต่าง HMI จะประกอบไปด้วย

- ตัวนับเวลา (Time) โดยจะนับเป็นวินาทีเพิ่มขึ้นไปเรื่อย ๆ
- Flow Rate Input : ปริมาณของของเหลวที่ไหลเข้าถัง
- Flow Rate Output : ปริมาณของของเหลวที่ไหลออกจากถัง
- Total : ปริมาณของของเหลวที่มีอยู่ในถัง
- Last Update : ปริมาณน้ำในถังก่อนที่จะถูก Update ล่าสุด
- Reset Button : ปุ่มสำหรับให้ผู้ใช้กดเพื่อ Reset ค่าปริมาณของของเหลวในถัง
- Count of Reset : จำนวนครั้งที่ผู้ใช้ทำการกดปุ่ม Reset



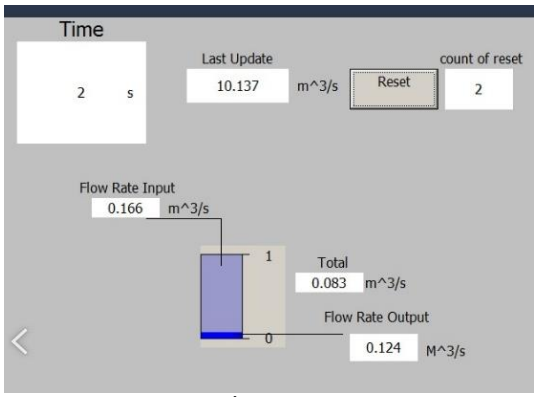
รูปที่ 45: HMI1

HMI เมื่อเริ่มรันโปรแกรมผ่านไป 5 วินาที จะสังเกตได้ว่า ค่า Flow Rate Input และ Flow Rate Output จะแสดงอัตราการไหลออกมาเป็นตัวเลข พร้อมกับปริมาณของของเหลวที่อยู่ในถัง(Total)จะเพิ่มขึ้น



รูปที่ 46: HMI2

HMI เมื่อเริ่มรันโปรแกรมผ่านไป 23 วินาที จะสังเกตได้ว่าปริมาณของของเหลวที่อยู่ในถังจะเพิ่มขึ้น



รูปที่ 47: HMI3

เมื่อมีการกดปุ่ม Reset โดยโปรแกรมจะแสดงจำนวนการกดปุ่ม Reset (Count of reset) ว่าผู้ใช้ได้สั่งให้ Reset ปริมาณของเหลวในถัง (Total) ไปเป็นจำนวนกี่ครั้งแล้ว

- **Question:**

1. What is the obvious additional feature for better control of this totalizer?
   ANS: ใช้ OB35 ในการนับเวลา 10 วินาที แทนการใช้ On-Delay Timer เนื่องจาก OB35 มีการทำงานทุกๆ 1 วินาที อยู่แล้ว

- **Summary:**

โปรแกรม totalizer เป็นโปรแกรมสำหรับวัดปริมาณน้ำในแทงค์ทั้งหมดในแต่ละวัน ซึ่งจะมีการ reset ปริมาณน้ำเป็น 0 ในทุก ๆ 24 ชั่วโมง และถ้าหากภายใน 10 วินาที เรากดปุ่ม reset ได้มากกว่า 20 ครั้ง ปริมาณน้ำก็จะถูก reset เป็น 0 ด้วยเช่นกัน

เราจะใช้ input เป็นสัญญาณ analog (4-20 mA) แทนช่วงอัตราการไหลของน้ำ 0-1.1 m3/s ซึ่งเราจะสามารถหาปริมาตรน้ำทั้งหมดในแทงค์ (current totalizer) ได้จากการใช้สูตรคือ ปริมาตรน้ำทั้งหมด = ปริมาตรน้ำที่เติมเข้ามา – ปริมาตรน้ำที่ไหลออก โดยการเขียนโปรแกรมเพื่อหาปริมาตรน้ำทั้งหมดนี้เราได้เขียนไว้ใน source code ภาษา SCL ซึ่งจะมีการเรียกใช้ควบคู่กันในการเขียนโปรแกรม LADDER

ในส่วนของการสั่ง reset ปริมาณน้ำให้เป็น 0 นั้น มี 2 เงื่อนไข ดังนี้

1) reset ปริมาณน้ำทุก ๆ 24 ชั่วโมง จะทำได้โดยสร้างการนับเวลาเพิ่มทุก ๆ 1 วินาทีไว้ แล้วหากเวลาที่นับได้ มากกว่า 86,400 วินาที หรือ 24 ชั่วโมง ระบบถึงจะทำการ reset ปริมาณน้ำ
2) reset ด้วยการกดปุ่มให้ได้มากกว่า 20 ครั้ง ภายใน 10 วินาที ซึ่งเงื่อนไขข้อนี้หลัก ๆ เราเขียนโปรแกรมไว้ใน LADDER คือ สร้างปุ่ม reset ขึ้นมา และมีการใช้ timer นับเวลา ในกรณีที่กดปุ่ม reset ได้น้อยกว่า 21 ครั้ง จะไม่มีการ reset ระบบเกิดขึ้น แต่กรณีที่มีการกดปุ่ม reset ได้มากกว่า 20 ครั้ง ระบบทั้งหมดก็จะถูก reset และค่า current tatalizer ก็จะย้ายไปแสดงค่าที่ last value of totalizer แทน