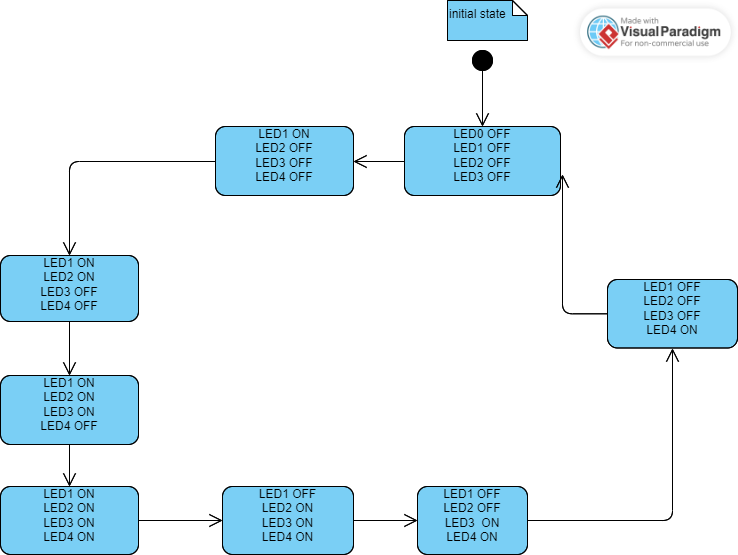
Project Title: Led Sequence V 1.0

Name: Basel Nagy

Description:

1. ***Hardware Requirements***
   1. *Four LEDs (LED0, LED1, LED2, LED3)*
   2. One button (BUTTON0)
2. ***Software Requirements***
   1. Initially, all LEDs are OFF
   2. Once BUTTON0 is pressed, LED0 will be ON
   3. Each press further will make another LED is ON
   4. At the fifth press, LED0 will changed to be OFF
   5. Each press further will make only one LED is OFF
   6. This will be repeated forever
   7. The sequence is described below
      1. Initially (OFF, OFF, OFF, OFF)
      2. Press 1 (ON, OFF, OFF, OFF)
      3. Press 2 (ON, ON, OFF, OFF)
      4. Press 3 (ON, ON, ON, OFF)
      5. Press 4 (ON, ON, ON, ON)
      6. Press 5 (OFF, ON, ON, ON)
      7. Press 6 (OFF, OFF, ON, ON)
      8. Press 7 (OFF, OFF, OFF, ON)
      9. Press 8 (OFF, OFF, OFF, OFF)
      10. Press 9 (ON, OFF, OFF, OFF)

State machine:



Layered architecture:

APP

HAL

MCAL

APPLICATION

LED BUTTON

DIO DRIVER

Project Modules APIs:

DIO DRIVER:

/\*typedef\*/

typedef enum DIO\_PORTS

{

porta, portb, portc, portd

} DIO\_PORTS;

typedef enum DIO\_PINS

{

pin0, pin1, pin2, pin3, pin4, pin5, pin6, pin7

} DIO\_PINS;

typedef enum PIN\_DIRECTION

{

INPUT,

OUTPUT

} PIN\_DIRECTION;

typedef enum PIN\_STATE

{

LOW,

HIGH

} PIN\_STATE;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* APIs PROTOTYPES \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

STD\_return DIO\_INIT (DIO\_PORTS port, DIO\_PINS pin, PIN\_DIRECTION direction);

STD\_return DIO\_WRITE\_PIN (DIO\_PORTS port, DIO\_PINS pin, PIN\_STATE state);

STD\_return DIO\_READ\_PIN (DIO\_PORTS port, DIO\_PINS pin, uint8\_t\* vale);

LED APIs:

typedef struct LED

{

DIO\_PORTS port;

DIO\_PINS pin;

} LED;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* APIs PROTOTYPES \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

STD\_return LED\_INIT (LED\* led);

STD\_return LED\_ON (DIO\_PORTS, DIO\_PINS);

STD\_return LED\_OFF (DIO\_PORTS,DIO\_PINS);

BUTTON APIs:

typedef enum EN\_BUTTON\_STATE\_t {PUSHED,RELEASED} EN\_BUTTON\_STATE\_t;

typedef struct BUTTON

{

DIO\_PORTS port;

DIO\_PINS pin;

} BUTTON;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* APIs PROTOTYPES \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

STD\_return BUTTON\_INIT (BUTTON\* button);

STD\_return BUTTON\_READ (DIO\_PORTS port, DIO\_PINS pin, EN\_BUTTON\_STATE\_t\* value);