(06)

Digital Input Output Driver Design Document

1. Description:  
   **This is a software driver for Digital Input Output Peripheral of Atmega 32 Microcontroller, this driver was developed by Anas Ebrahim at 25/3/2016 under the supervision of Eng.Mohammad Hassan and Eng.Walid El-Hennawy in the Software Engineering Course.  
   The driver provides the general APIs and Macros needed to use the 32 Digital Input output Pins of the Microcontroller**
2. Driver Architecture:  
   **The driver lies on the MCAL Layer and contains 3 header files  
   1-DIO\_Interface.h which contains the Functions/APIs Prototypes and variable like macros the user can use   
   2-DIO\_Config.h which contains the configuration the user can choose to be the initial directions and values of the input output pins.  
   3-DIO\_private.h which contains macros that is used only inside the driver.  
   And one source file   
   DIO\_Prog.c which contains the Implementation of the driver APIs.**  
   **The Driver also uses (util.h) & (types.h) library from the libraries layer.**
3. Configurations:  
   **The user is required to configure the Initial Direction and Values for each Pin.  
   - The options of the direction should be either (DIO\_u8OUTPUT) or (DIO\_u8INPUT)   
   - The options of the Value should be either (DIO\_u8HIGH) or (DIO\_u8LOW)  
   - The user should choose pins from Range DIO\_u8PIN0 to DIO\_u8PIN31 .**
4. APIs   
   1-Public: (DIO\_00\_06\_01)  
   **a- DIO\_VoidInit(void):** Initialization function that assigns the Initial Direction and values of the DIO Pins that is configured by the user.  
     
    **b- DIO\_u8ReadPinVal(u8 Copy\_u8PinIdx, u8\* Copy\_u8PtrToVal):**  
   Read Input Pin Value function which Takes a Pin index and a pointer to save the pin value at and returns the function state, U8Error if the Index is out of boundary or if the pin is output and u8OK otherwise.  
     
     
   **c- DIO\_u8WritePinVal(u8 Copy\_u8PinIdx, u8 Copy\_u8PinVal):**  
   Write Output Pin Value function which takes a Pin index and the pin value and returns the function state, U8Error if the Index is out of boundary or if the pin is input and u8OK otherwise.  
     
   **d- DIO\_u8ReadPortVal(u8 Copy\_u8PortIdx, u8\* Copy\_u8PtrToVal):**  
   Read Input Port Value function which takes a Port index and a pointer to save the port value at and returns the function state, U8Error if the Index is out of boundary or if the port is output and u8OK otherwise.  
     
   **e- DIO\_u8WritePortVal(u8 Copy\_u8PortIdx, u8 Copy\_u8PortVal):**  
   Write output Port Value function which takes a Port index and the port value and returns the function state, U8Error if the Index is out of boundary or if the port is input and u8OK otherwise   
     
   **f- DIO\_u8WritePinDir(u8 Copy\_u8PinIdx, u8 Copy\_u8PinDir):**  
   Write Pin Direction function which takes a Pin index and the pin direction and returns the function state, U8Error if the Index is out of boundary and u8OK otherwise  
     
   **g- DIO\_u8WritePortDir(u8 Copy\_u8PortIdx, u8 Copy\_u8PortDir)**  
   Write Port Direction function which Takes a Port index and the port direction and returns the function state, U8Error if the Index is out of boundary and u8OK otherwise  
     
   **h- DIO\_u8ReadPinDir(u8 Copy\_u8PinIdx, u8\* Copy\_u8PtrToDir)**  
   Read Pin Direction function which Takes a Pin index and a pointer to the pin direction and returns the function state, U8Error if the Index is out of boundary and u8OK otherwise  
     
   **i- DIO\_u8ReadPortDir(u8 Copy\_u8PortIdx, u8\* Copy\_u8PtrToDir)**  
   Read Port Direction function which Takes a Port index and a pointer to the port direction and returns the function state, U8Error if the Index is out of boundary and u8OK otherwise  
     
   2- Private:  
   **conc(bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7)**  
   Concatenation function like macro which takes 8 bit binary values and concatenates them into one byte.
5. Shared Variables   
   **There is no shared variables in the driver**
6. Integration constrains  
   **1- The Pin is not reserved for another peripheral that is used in the application   
   2-Choosing the proper configuration of the pin with the external hardware components, the wrong direction of the pin may damage the Microcontroller  
   3-choosing the proper initial value of the output pin that is required to the attached module to the pin**
7. Hardware constrains  
   **1-All the not used pins should be input  
   2-The Pins can drive up to 15mA per pin   
   3- The sum of all pins generated current per port should not exceed 100mA.**