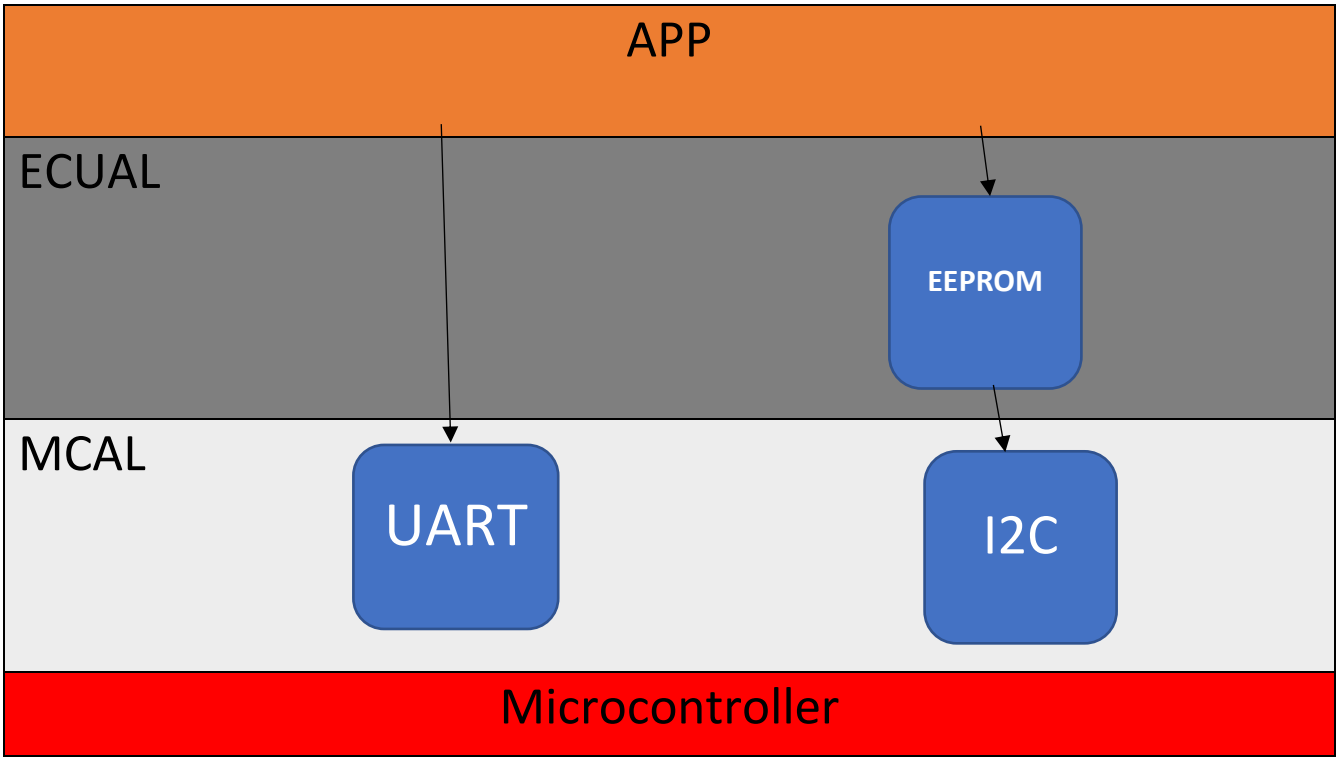
Layers:



APP:

```
void APPInit(void);
void APP_GetState(void);
void APP_UpdateState(void);
```

EEPROM:

```
void ECUAL_EEPROMInit(void);
void ECUAL_EEPROMWrite(uint8_t EEPROM_PageNum, uint8_t u8_MEMAddress , uint8_t u8_Data);
void ECUAL_EEPROMRead(uint8_t EEPROM_PageNum, uint8_t u8_MEMAddress , uint8_t *u8_DataRead);
```

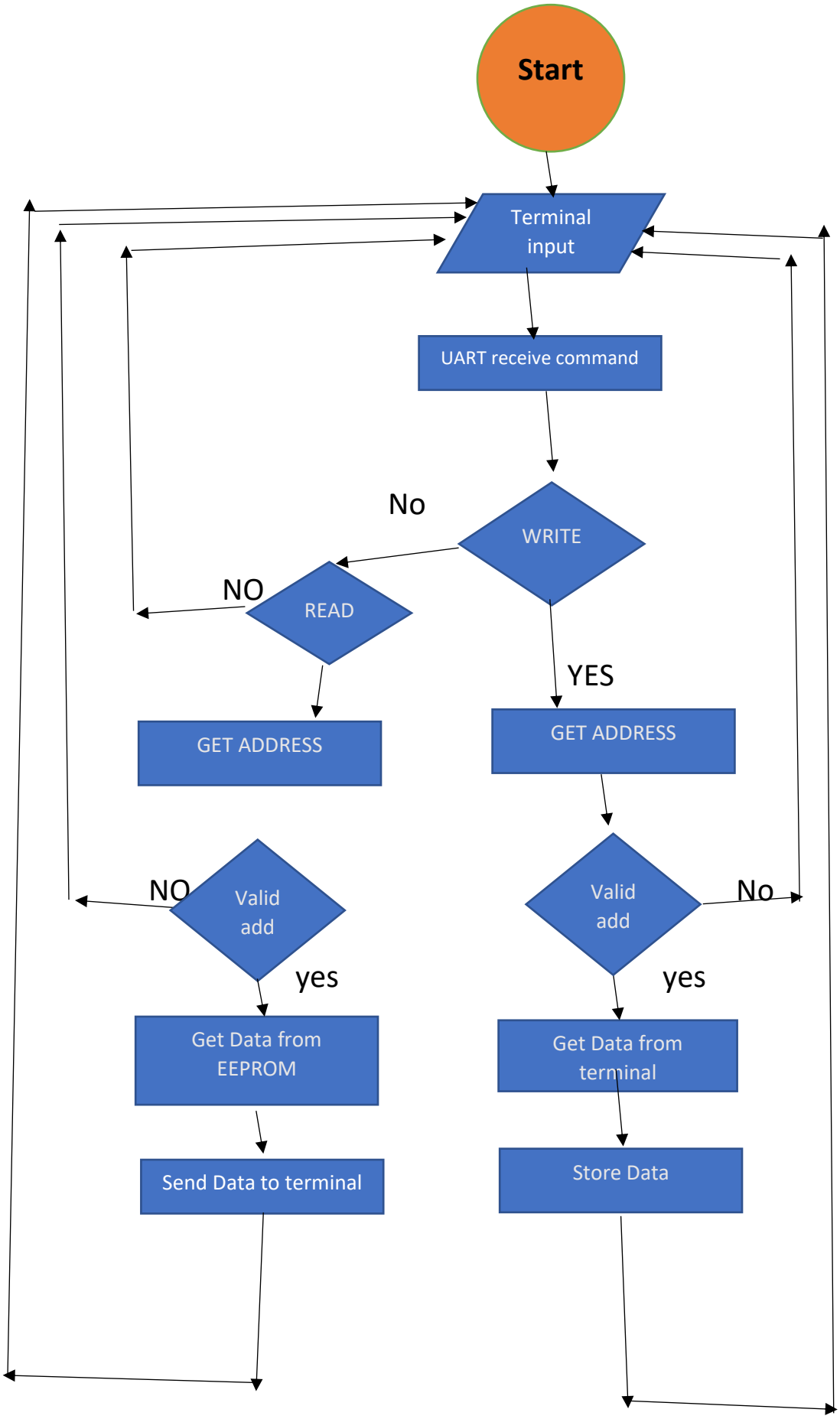
UART:

```
UART_ERROR_t USART_Init(void);
UART_ERROR_t USART_Transmit(uint16_t u16_data);
UART_ERROR_t USART_Receive (uint16_t *Ptr_to_val);
UART_ERROR_t USART_Transmit_Packet(uint8_t *pu8_data,uint8_t SIZE);
UART_ERROR_t USART_receive_Packet(uint8_t *pu8_data,uint8_t u8_SIZE);
UART_ERROR_t USART_Transmit_INTDriven(uint8_t *pu8_data,void (*ptr)(void));
UART_ERROR_t USART_Receive_INTDriven(uint8_t *pu8_data,void (*ptr)(void));
```

I2C:

```
void MCAL_I2CMasterInit(Master_config_t *MasterConfig);
void MCAL_I2CSlaveInit(uint8_t u8_SlaveAddress);
I2C_status_codes MCAL_I2CMaterEstablishComm(uint8_t u8_slaveAddress);
I2C_status_codes MCAL_I2CMasterRepeatedStart(uint8_t u8_slaveAddress);
I2C_status_codes MCAL_I2CMasterWrightData(uint8_t* u8_Data);
I2C_status_codes MCAL_I2CMasterReadsendACK(uint8_t *readData);
I2C_status_codes MCAL_I2CMasterReadsendNACK(uint8_t *readData);
void MCAL_I2CMasterStopCondition(void);
I2C_slave_status_codes MCAL_I2CSlaveListen(void);
I2C_slave_status_codes MCAL_I2CSlaveTransmit(uint8_t *u8_DataSend);
I2C_slave_status_codes MCAL_I2CSlaveReceive(uint8_t *ReceivedData);
```

Flow chart:


Start
Terminal input
UART receive command
WRITE
No
READ
NO
YES
GET ADDRESS
GET ADDRESS
Valid add
NO
yes
Valid add
No
yes
Get Data from EEPROM
Get Data from terminal
Send Data to terminal
Store Data