2 API Capability

M2MX60P API consists of GPIO, PWM, and UART libraries to support related functions.

3 Use M2MX60P API in user program

3.1 Copy library files to the directory of user program

libgpio.jar
libpwm.jar
libuart.jar

3.2 Edit Android.mk of user program

```
include $(CLEAR_VARS)
LOCAL_STATIC_JAVA_LIBRARIES := libgpio libpwm libuart
LOCAL_PACKAGE_NAME := .....
.....
include $(BUILD_PACKAGE)
##################################################
include $(CLEAR_VARS)
LOCAL_PREBUILT_STATIC_JAVA_LIBRARIES := libgpio.jar libpwm.jar libuart.jar
include $(BUILD_MULTI_PREBUILT)
```

3.3 Edit JAVA source of user program

Refer to the following chapters

4 GPIO library usage

4.1 Import GPIO library at the beginning of JAVA source code

```
import com.zzzapi.gpio.*;
```

4.2 Initiate the GPIO-lib object

```
gpio gpio_tool = new gpio();
```

4.3 Write to pre-defined GPIO pins

```
int value = 0; //0 or 1
gpio_tool.set(gpio.PIN.POWERLED, gpio_value);
gpio_tool.set(gpio.PIN.DRAWER_1_OPEN, value);
gpio_tool.set(gpio.PIN.DRAWER_2_OPEN, value);
gpio_tool.set(gpio.PIN.GPIO2, value);
gpio_tool.set(gpio.PIN.GPIO3, value);
gpio_tool.set(gpio.PIN.VDRAW_ONOFF, value);
```

4.4 Read from pre-defined GPIO pins

```
int value;
value = gpio_tool.get(gpio.PIN.CHECK_VDRAWER_1_OPEN);
value = gpio_tool.get(gpio.PIN.CHECK_VDRAWER_2_OPEN);
value = gpio_tool.get(gpio.PIN.DETECT_POWER_FAIL);
```

5 PWM library usage

5.1 Import PWM library at the beginning of JAVA source code

```
import com.zzzapi.pwm.*;
```

5.2 Initiate the PWM-lib object

```
pwm    pwm_tool = new pwm();
```

5.3 Start buzzer at specified frequency

```
int frequency = 1234; //any positive integer
pwm_tool.buzzerStart(frequency);
```

5.4 Stop buzzer

```
pwm_tool.buzzerStart(0);
```

6 UART library usage

6.1 Import UART library at the beginning of JAVA source code

```
import com.zzzapi.uart.*;
```

6.2 Initiate the UART-lib object

```
uart uart_tool = new uart();
```

6.3 COM port config

```
int com_port = 0; //COM port: 0~3
int baud = 115200; //baud rate
int data_bits = 8; //number of data bits
int stop_bits = 1; //number of stop bits
uart_tool.config(com_port, baud, data_bits, stop_bits);
```

6.4 Write message to COM port

```
int com_port = 0; //COM port: 0~3
String msg = new String("Hello world!");
uart_tool.write(com_port, msg);
```

6.5 Show message from COM port

1. Locate a view object on UI layout.
2. Register the view to UART-lib.

Then this view can show message from the specified COM port.

Code sample:

```
int com_port = 0; //COM port: 0~3
TextView mUartInput = (TextView)findViewById(R.id.uartInput);
//resource ID (R.id.uartInput for example here) must be referred to UI layout
uart_tool.showInputOnView(com_port, mUartInput);
```