M66 OpenCPU Solution Presentation

October, 2014

Outline

- **OpenCPU Summary**
- Open Resources
- Advantages
- Software Architecture
- What's New?
- Development Requirements

OpenCPU Summary

OpenCPU is an embedded development solution for M2M field. Based on it, customer can conveniently design the embedded applications. It enables customer to create innovative application and download it directly into Quectel module to run. In the OpenCPU solution, GSM/GPRS module acts as a main processor. So, GSM/GPRS module with OpenCPU solution facilitates customer's product design and accelerates the application development.

M66 OpenCPU Module is a powerful functional Quad-band GSM/GPS module in LCC castellation packaging. It has a compact and tiny dimension of 15.8×17.7×2.3mm. Besides Internet, telephony, SMS, FOTA, data storage options, it builds in advanced audio player, recorder, QuecLocator, eCall, and the new feature "Bluetooth", which make this module the best choice for the applications that have the strict requirements on extended functions and cost-effectiveness.

OpenCPU module can be widely used in M2M field, such as tracker & tracing, automotive, energy, etc.






Outline

- OpenCPU Summary
- Open Resources
- Advantages
- Software Architecture
- What’s New?
- Development Requirements

System Resource on M66 OpenCPU Module

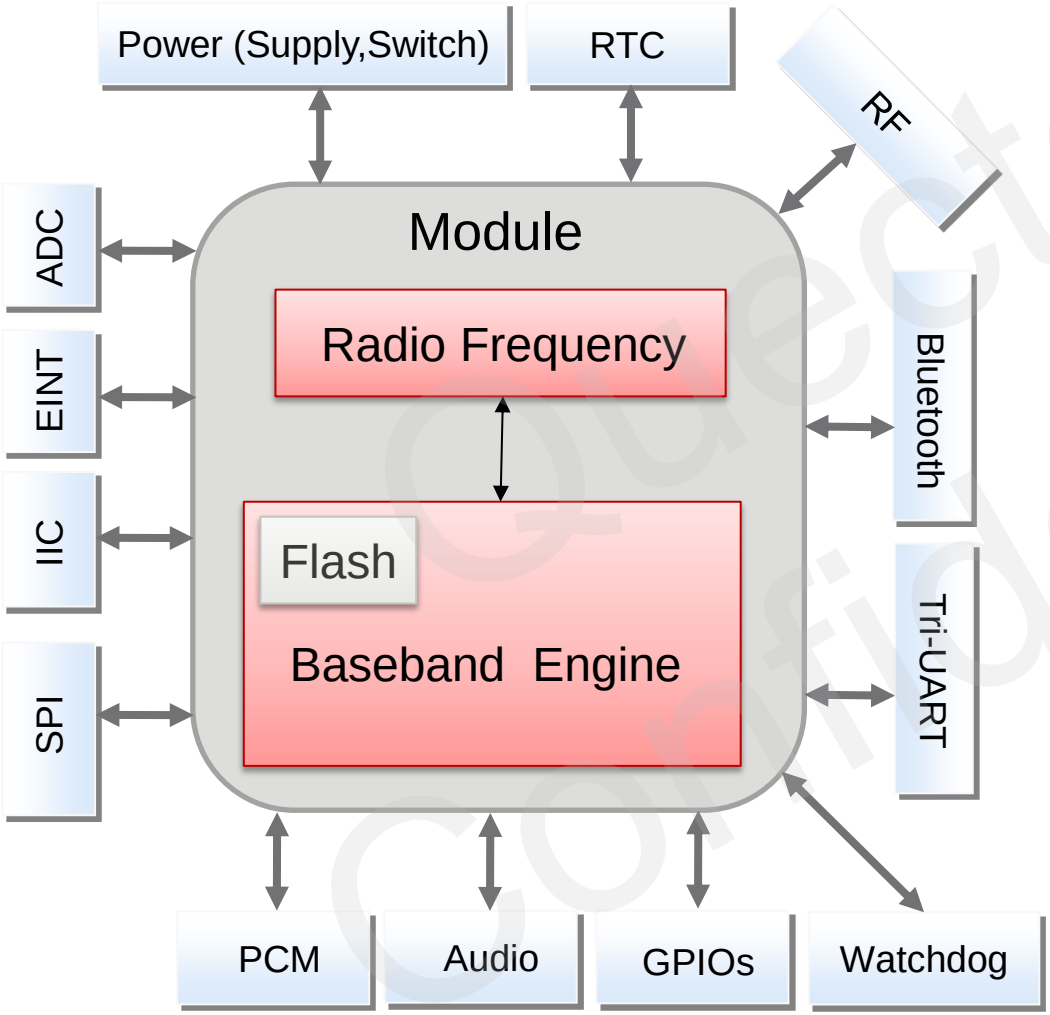
- **CPU**

  32-BIT ARM7EJ-S™ RISC 260MHz

- **MEMORY (4MB Flash + 4MB RAM)**
  - Code Region: 360KB space for App image bin
  - RAM: 100KB static memory and 500KB dynamic memory
  - UFS Region: 300KB space

Open Resources (2)

Hardware Architecture



Hardware Resource

- Power supply
- Power switch
- RTC
- 3 UART ports
- ADC
- PCM
- Audio interfaces (2 output channels and 1 input channel)
- Bluetooth
- GPIO interfaces
- PWM output interface
- EINT interfaces
- IIC interface
- SPI interface
- Watchdog

Open Resources (3)

Programmable Multiplexing Pins:

PIN No.	PIN NAME	MODE1 (default)	MODE2	MODE3	MODE4
16	PINNAME_NETLIGHT	NETLIGHT	GPIO	PWM_OUT	
19	PINNAME_DTR	DTR	GPIO	EINT	SIM_PRESENCE
20	PINNAME_RI	RI	GPIO	I²C_CLK	
21	PINNAME_DCD	DCD	GPIO	I²C_SDA	
22	PINNAME_C TS	CTS	GPIO		
23	PINNAME_RTS	RTS	GPIO		
28	PINNAME_RXD_AUX	RXD_AUX	GPIO		
29	PINNAME_TXD_AUX	TXD_AUX	GPIO		
30	PINNAME_PCM_CLK	PCM_CLK	GPIO	SPI_CS	
31	PINNAME_PCM_SYNC	PCM_SYNC	GPIO	SPI_MISO	
32	PINNAME_PCM_IN	PCM_IN	GPIO	SPI_CLK	
33	PINNAME_PCM_OUT	PCM_OUT	GPIO	SPI_MOSI	

Note: The interrupt response time is 50ms by default, and can be re-programmed to a bigger value in OpenCPU. However, it is strongly recommended that the interrupt frequency cannot be more than 3Hz for the sake of module stable working.

Outline

- OpenCPU Summary
- Open Resources
- **Advantages**
- Software architecture
- What's New?
- Development Requirements

Advantages – Low Cost, Fast Develop

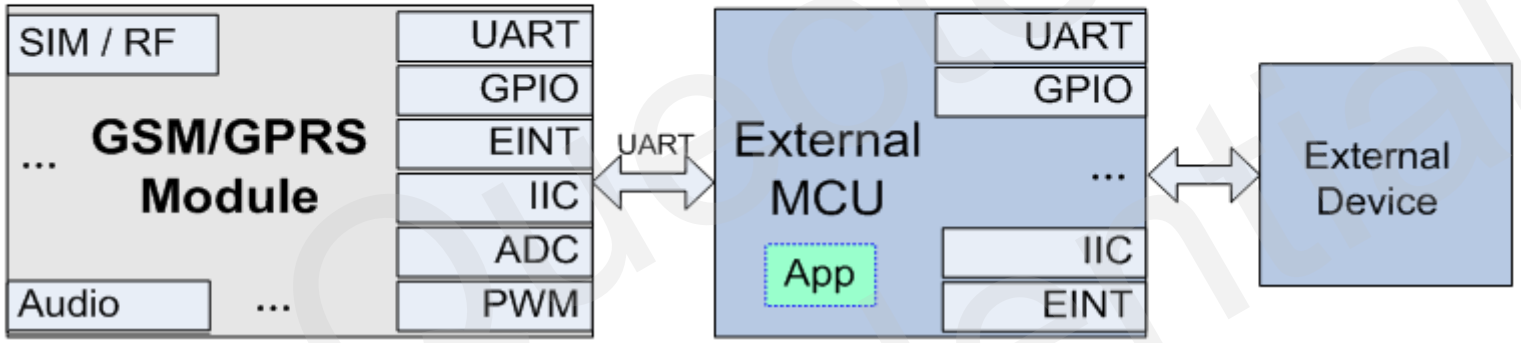
- Reduce product development time
- Simplify circuit design and reduce costs and power consumption
- Decrease the product's size
- Upgrade firmware remotely via OpenCPU FOTA
- Decrease the total cost and enhance the competitive advantages

Low Cost, Fast Develop

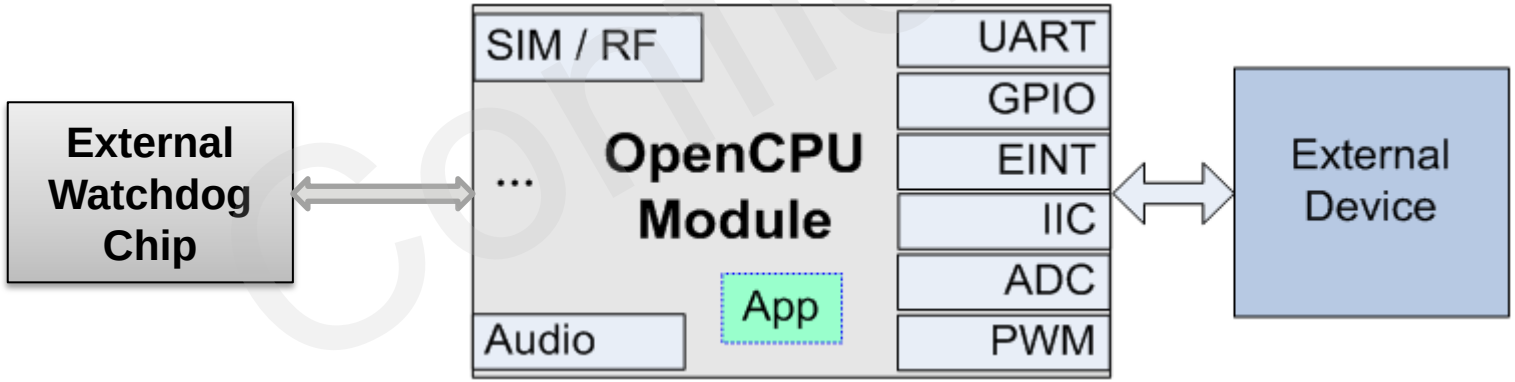
Advantages – Easy Hardware Design

Compared with traditional solution, OpenCPU solution can make hardware design easier for the developer. The figure below shows the differences between them.

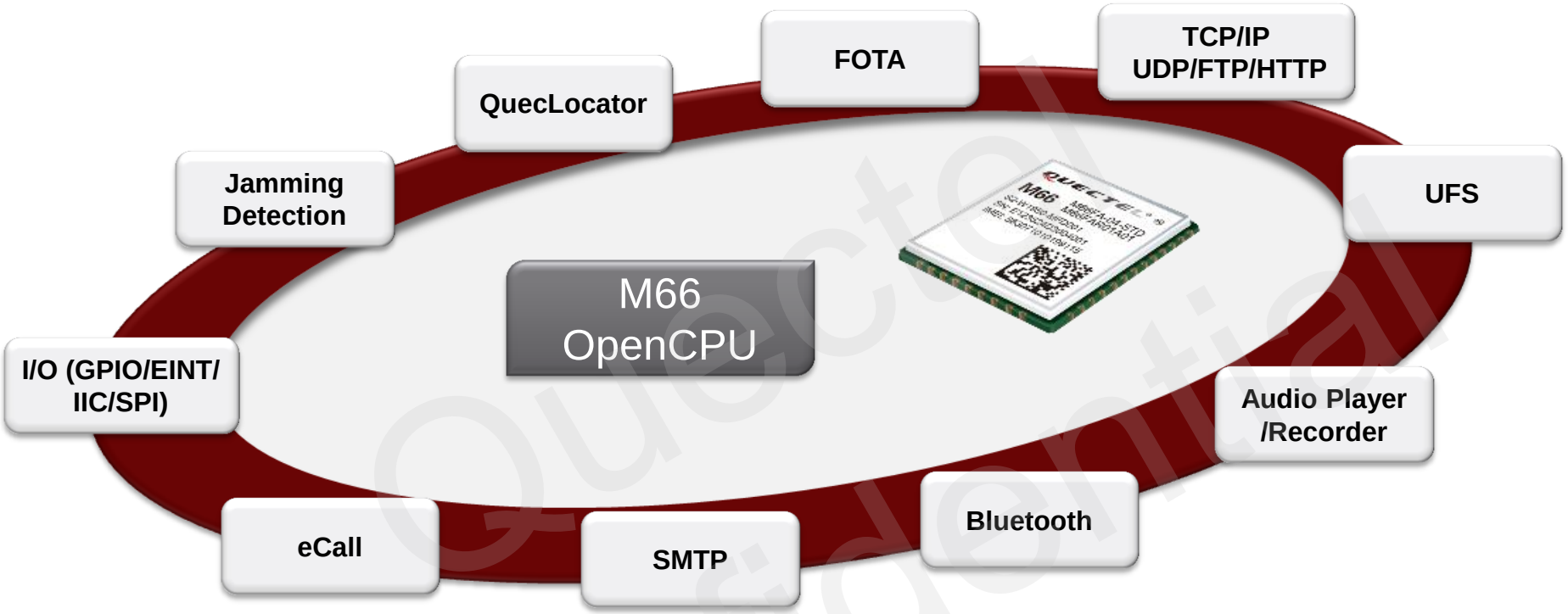
■ Traditional Solution



■ OpenCPU Solution



Advantages – Enhanced Technology



Quality Guarantee

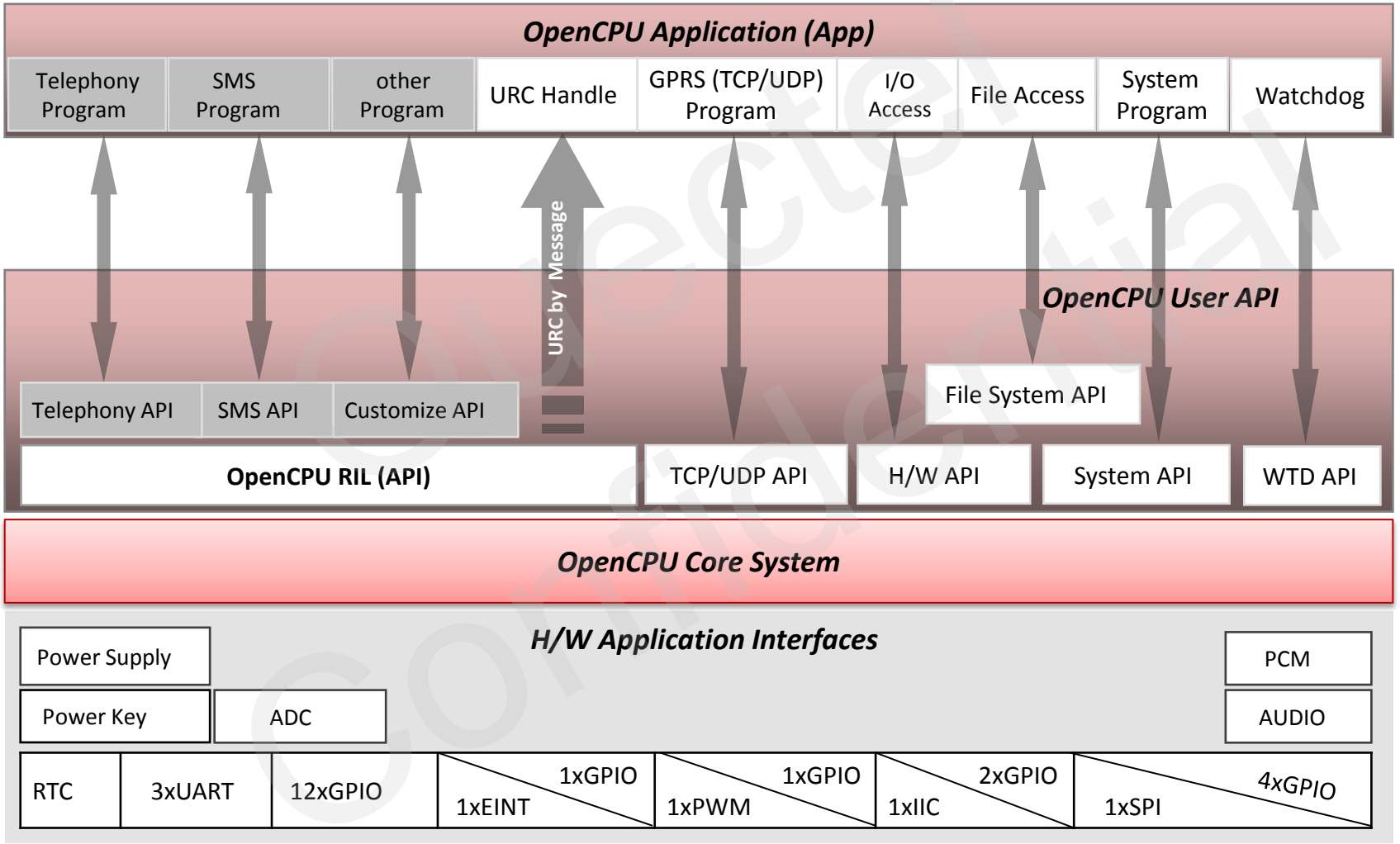
- Reliable network protocols
- Steady flash protection mechanism
- Superior audio algorithms
- Rich I/O interfaces
- Bluetooth 3.0

Outline

- OpenCPU Summary
- Open Resources
- Advantages
- **Software Architecture**
- What's New?
- Development Requirements

Software Architecture (1)

System software of OpenCPU consists of 3 layers: Core system, User API and Application. The following block diagram shows the software architecture of OpenCPU.



Software Architecture (2)

Core System

Core System is a combination of hardware and system software of GSM/GPRS module. It has a built-in ARM7EJ-S processor, and has been built over Nucleus operating system, which has the characteristics of micro-kernel, real-time, multi-tasking and etc.

OpenCPU RIL

OpenCPU RIL, a open source layer, is embedded into User API layer. With OpenCPU RIL, developer can simply call API to send AT commands and immediately get the response when API returns.

By default, AT commands related to SMS and telephone are wrapped in the RIL APIs. Developer can easily develop some new API functions to implement the AT commands according to the requirements.

OUTLINE

- OpenCPU Summary
- Open Resources
- Advantages
- Software Architecture
- **What's New?**
- Development Requirements

What's New? (1)

OpenCPU RIL Support

Compared with the previous OpenCPU solution, the new OpenCPU module provides RIL diver in user API layer, which makes software design easier. When the developer sends AT commands to core system, the responses are received and handled by RIL layer first, and then RIL returns the eventual data that application wants. And also the new OpenCPU provides the Open Source of RIL. The following C code demonstrates the differences in details.

- **New OpenCPU solution**

```
s32  RIL_NW_GetSignalQuality(u32* rssi, u32* ber)
{
        s32 retRes = 0;
        char strAT[] = "AT+CSQ\0";
        ST_CSQ_Reponse pCSQ_Reponse;

        //Here Sending "AT+CSQ" command
        retRes = Ql_RIL_SendATCmd(strAT,Ql_strlen(strAT), ATResponse_CSQ_Handler,(void*)&pCSQ_Reponse,0);

        //Now the response of "AT+CSQ" is just in pCSQ_Reponse, then you can get the CSQ value
        if(RIL_AT_SUCCESS == retRes)
        {
                *rssi = pCSQ_Reponse.rssi;
                *ber = pCSQ_Reponse.ber;
        }
        return retRes;
}
```

What's New? (2)

➢ The previous OpenCPU solution

```
//Here, sending AT+CSQ
void SendAtCmd()
{
   s32 ret;
   ret = Ql_SendToModem(ql_md_port1, (u8*)"AT+CSQ\r", Ql_strlen("AT+CSQ\r"));
}
//Here, get the response of AT+CSQ in main loop body.
void ql_entry( )
{
   switch (g_cmd_idx)
   {
      case 1:// Echo mode off
       Ql_sprintf((char *)buffer, "ATE0\n");
   while(1)
   {
      Ql_GetEvent(&g_event);
      switch(g_event.eventType)
      {
         case EVENT_MODEMDATA:
         {
            //TODO: receive and handle data from CORE through virtual modem port
            PortData_Event* pPortEvt = (PortData_Event*)&g_event.eventData.modemdata_evt;
            // Here receive URC and the AT response
            break;
         } } }
}
```

What's New? (3)

GCC Compiler Support

- M66 OpenCPU supports free-of-charge GCC compiler (Sorcery CodeBench Lite ARM EABI).
- The previous OpenCPU solution uses ARM RVCT as compiler.

IDE Support

M66 OpenCPU recommends two sets of tools to manage codes and compile program. One is command-line + Source Insight; the other is Eclipse.

■ Easy to Configure the Initial Status of GPIO

M66 OpenCPU provides a very simple method to configure the initial status of GPIOs when these GPIOs need to be initialized after the early boot time. For example, GPIOs are used to control the power supply of peripheral circuit.

Here is an example code to configure the initial status of NETLIGHT pin. When the module powers up, NETLIGHT pin will be initialized to "output", low level" and "pull-down inside module".

```
/*------------------------------------------------------------------------------------------------------------------------------------------
Function Name     Pin Name                Direction(In or Out)          Level              Pull Selection(Down or Up)
------------------------------------------------------------------------------------------------------------------------------------------*/
GPIO_ITEM (  PINNAME_NETLIGHT,      PINDIRECTION_OUT,      PINLEVEL_LOW,      PINPULLSEL_PULLDOWN)
```

Besides, developer may call the GPIO-related APIs to reprogram the GPIO pin. For example, calling Ql_GPIO_SetLevel() may change the level of GPIO pin.

Easy to Add New Task (Thread)

Developer can simply follow the procedures below to add a task in "custom_task_cfg.h" file to define a new task.

- **Task Id Name:** Task Id Name is a totally customized name. Developer can define the name, and the system will automatically define and assign the value.
- **Task Stack Size**: The range of task stack size is from 1KB to 10KB. If there are any files operations in the task, the task size must be set to at least 5KB, otherwise, the stack overflow probably happens.
- **Default Value2**: Developer does not specify the value and set it to the default definition.

```
/*-------------------------------------------------------------------------------------------------------------------------------------
        Task Entry function     Task Id Name     Task Stack Size(Bytes)     Default Value1     Default Value2
-------------------------------------------------------------------------------------------------------------------------------------*/
TASK_ITEM (Proc_main_task,      main_task_id,        10*1024,          DEFAULT_VALUE1,    DEFAULT_VALUE2)
```

Programmable Power Key Pin for App

Here is an example on how to power on or power off the module by Power Key pin. It is very easy to program Power Key pin for the developer.

```
static const ST_PowerKeyCfg pwrkeyCfg = {
{
   TRUE,  // working mode for power-on on PWRKEY pin
   /*
   Module automatically powers on when feeding a low level to POWER_KEY pin.
   When setting to FALSE, the callback that Ql_PwrKey_Register registers will be trigged. Application must
   call Ql_LockPower () to lock power supply, or module will lose power when the level of PWRKEY pin goes
   high.
   */
   TRUE,  // working mode for power-off on PWRKEY pin
   /*
   Module automatically powers off when feeding a low level to POWER_KEY pin.
   When setting  to FALSE, the callback that Ql PwrKey_Register registers will be trigged.
   Application may do post processing before switching  off the module.
   */
};
```

Outline

- OpenCPU Summary
- Open Resources
- Advantages
- Software Architecture
- What's New?
- **Development Requirements**

Host System Requirements

The following host Operating Systems and architectures are supported:

- Microsoft Windows XP (SP1 or later)
- Windows Vista
- Windows 7 systems using IA32, AMD64, and Intel 64 processors.

Compiler & IDE Requirements

- GCC Compiler (Sorcery CodeBench Lite for ARM EABI)
- IDE: Eclipse (optional)

Development Requirements (2)

Programming Language Requirement

- Basic C-language programming knowledge

SDK and Other Requirements

- Quectel GSM/GPRS Module with OpenCPU Solution
- Quectel EVB
- OpenCPU SDK
- Firmware Download Tool (included in SDK)
- OpenCPU FOTA Package Tool (included in SDK)

Q&A…

Thank you