ARM Your Sensors

Prototyping a Wi-Fi ARMED IoT Sensor Node

August 27, 2018

Fred Eady

DesignNews

CEC CONTINUING EDUCATION CENTER

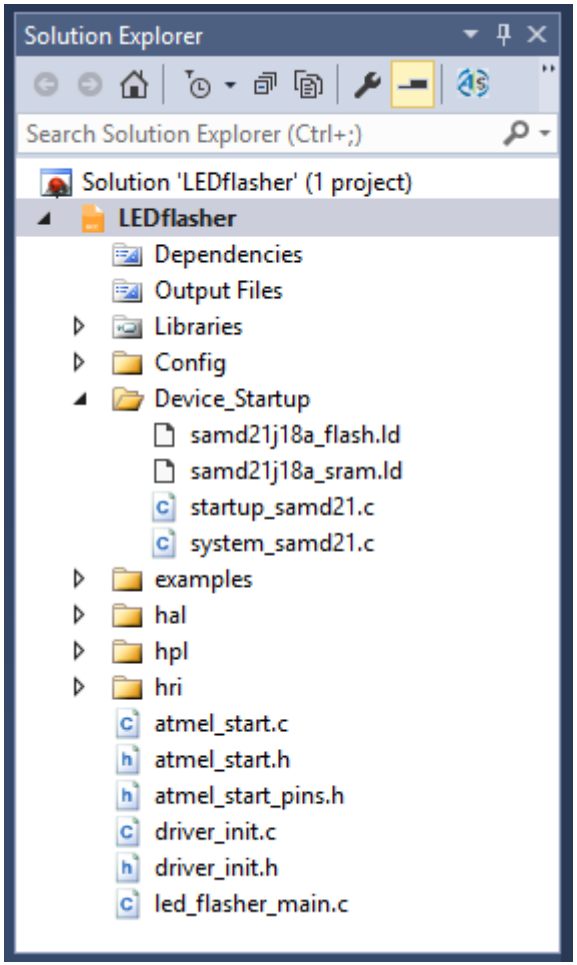
Presented by: Digi-Key ELECTRONICS

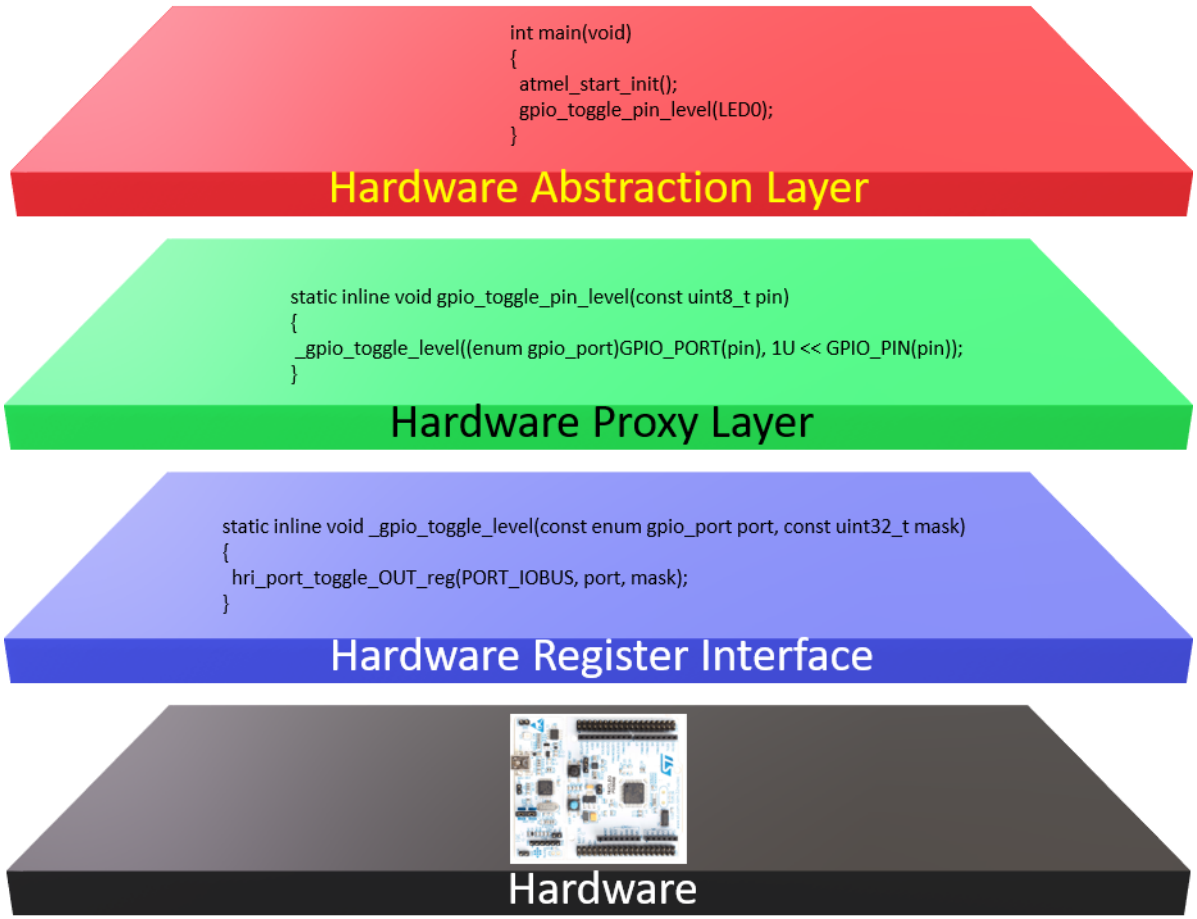
AGENDA

- ARMed Abstraction
- Lock and Load
- Firing UDP Datagrams
- Day 1 Summary

ARM Your Sensors

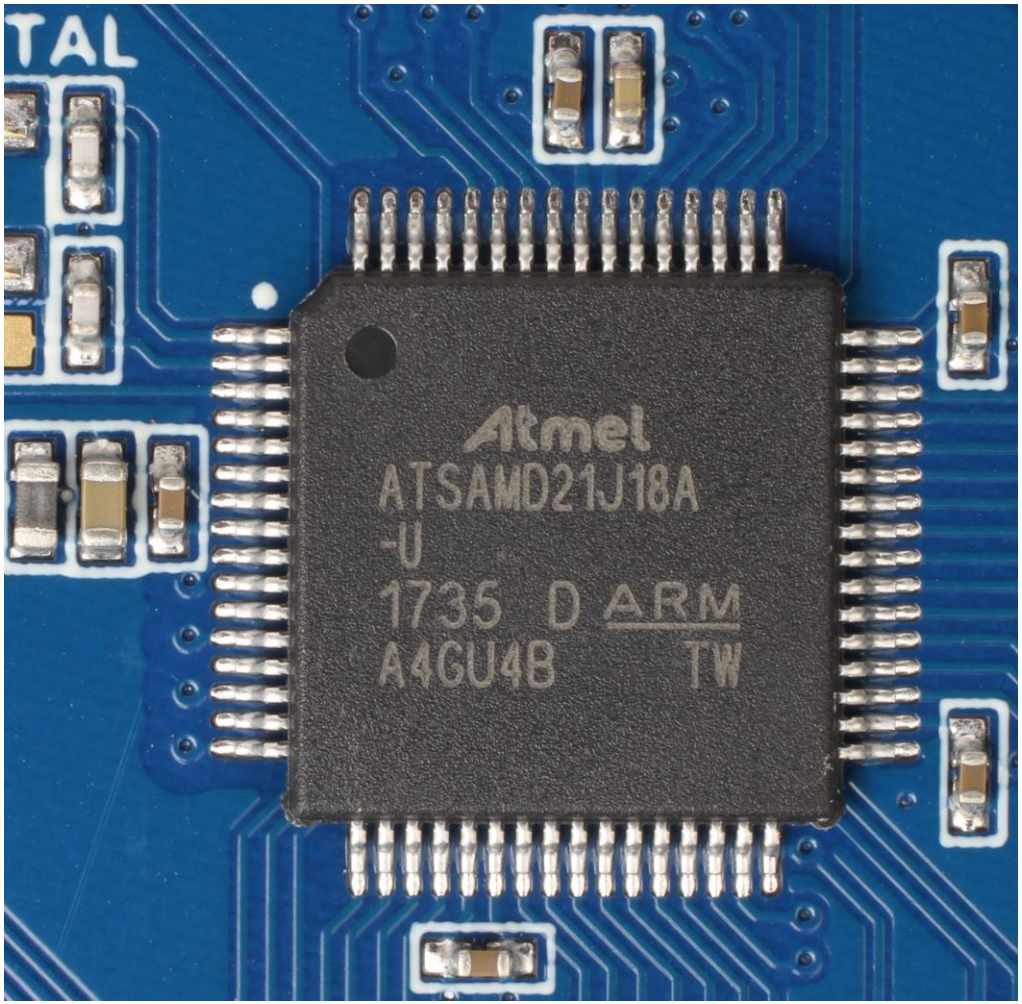
ARMed Abstraction – Advanced Software Framework (ASF)





ARM Your Sensors

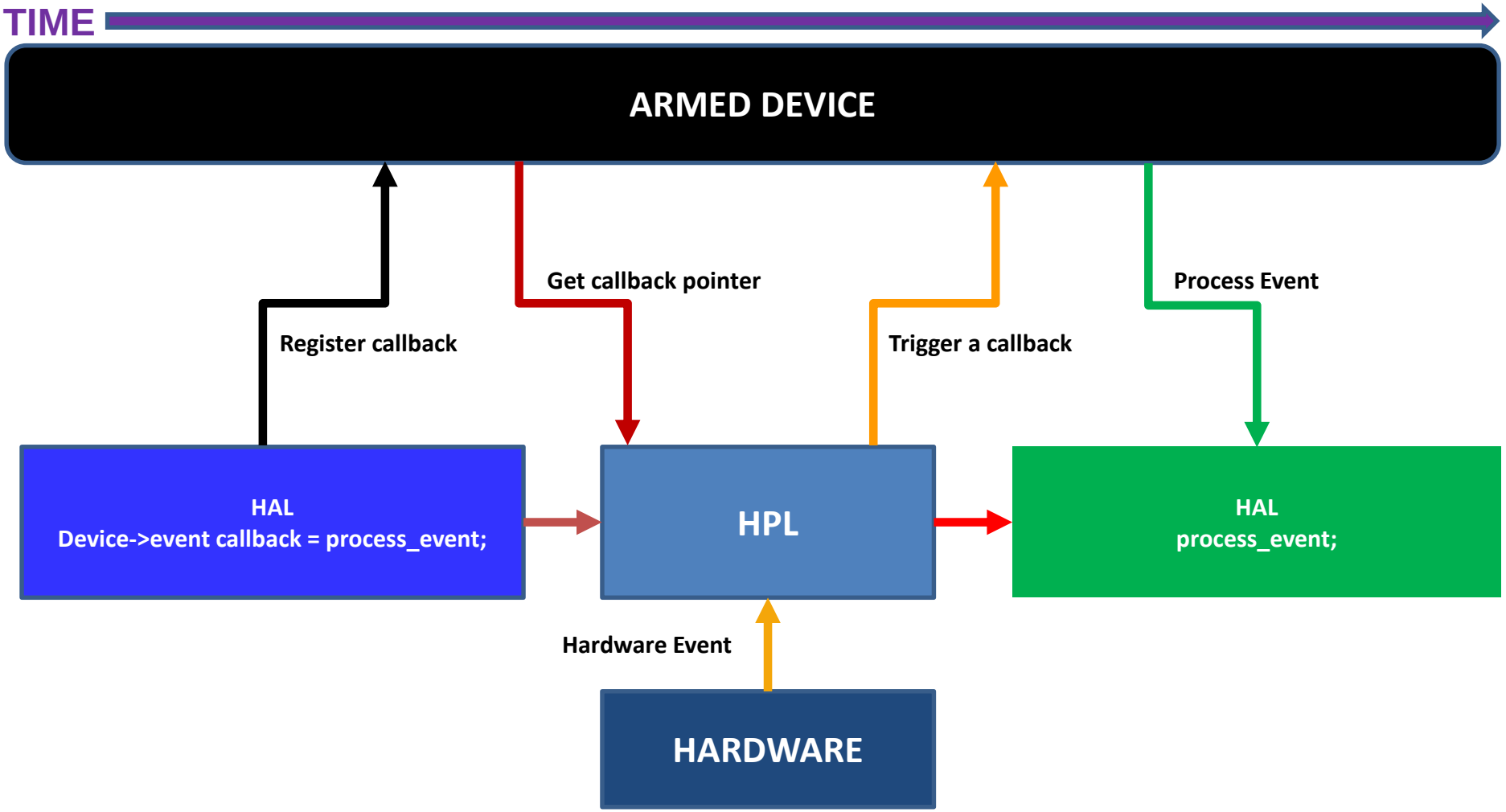
ARMed Abstraction – Advanced Software Framework (ASF)




```
struct _usart_device
{
  struct _usart_callbacks usart_cb
  void *hw;
};

_usart_init(void *const hw);
_usart_deinit(void *const hw);
_usart_transmit(void *const hw, uint8_t *const buf, uint32_t length);
```

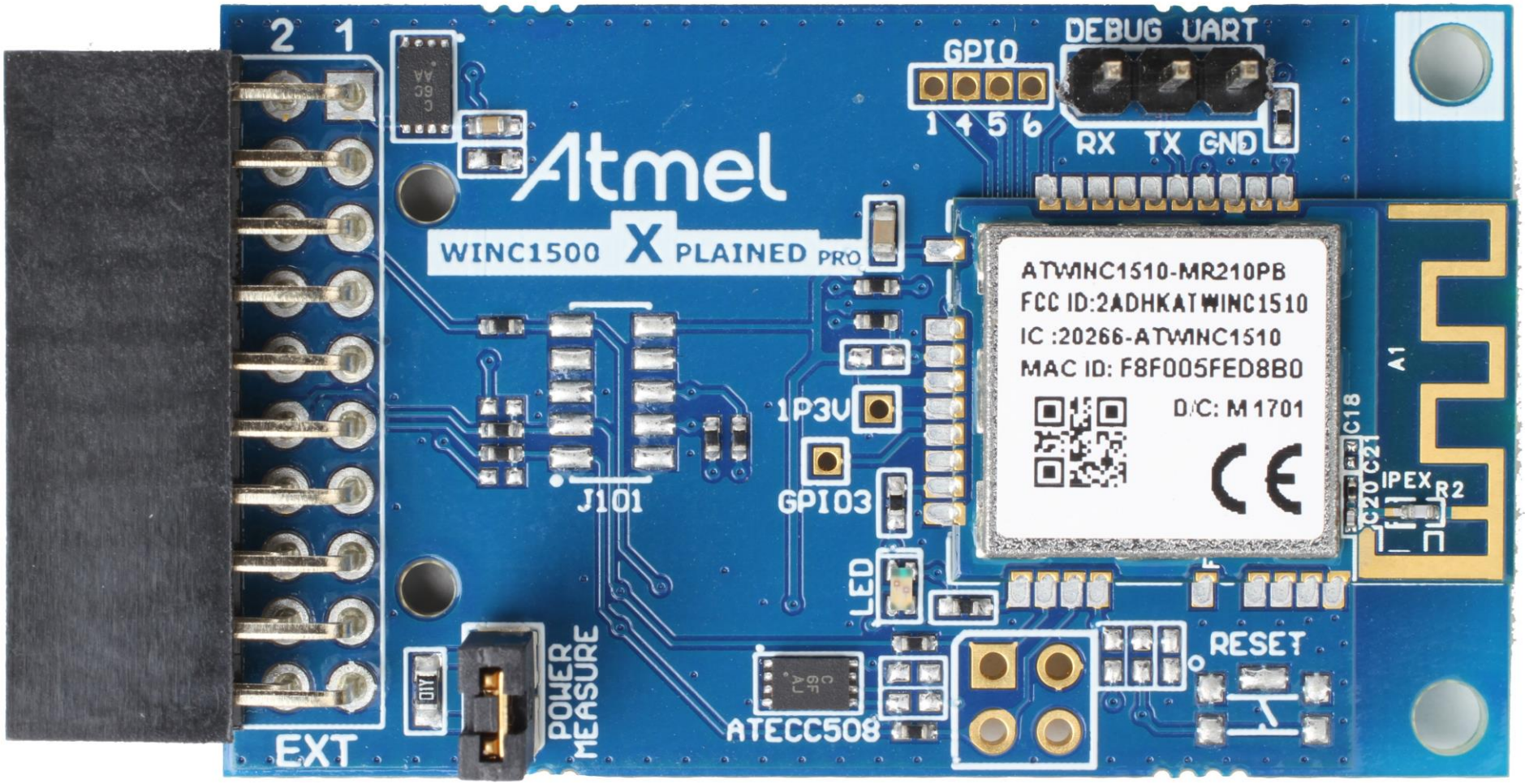
ARM Your Sensors

ARMed Abstraction – Advanced Software Framework (ASF)



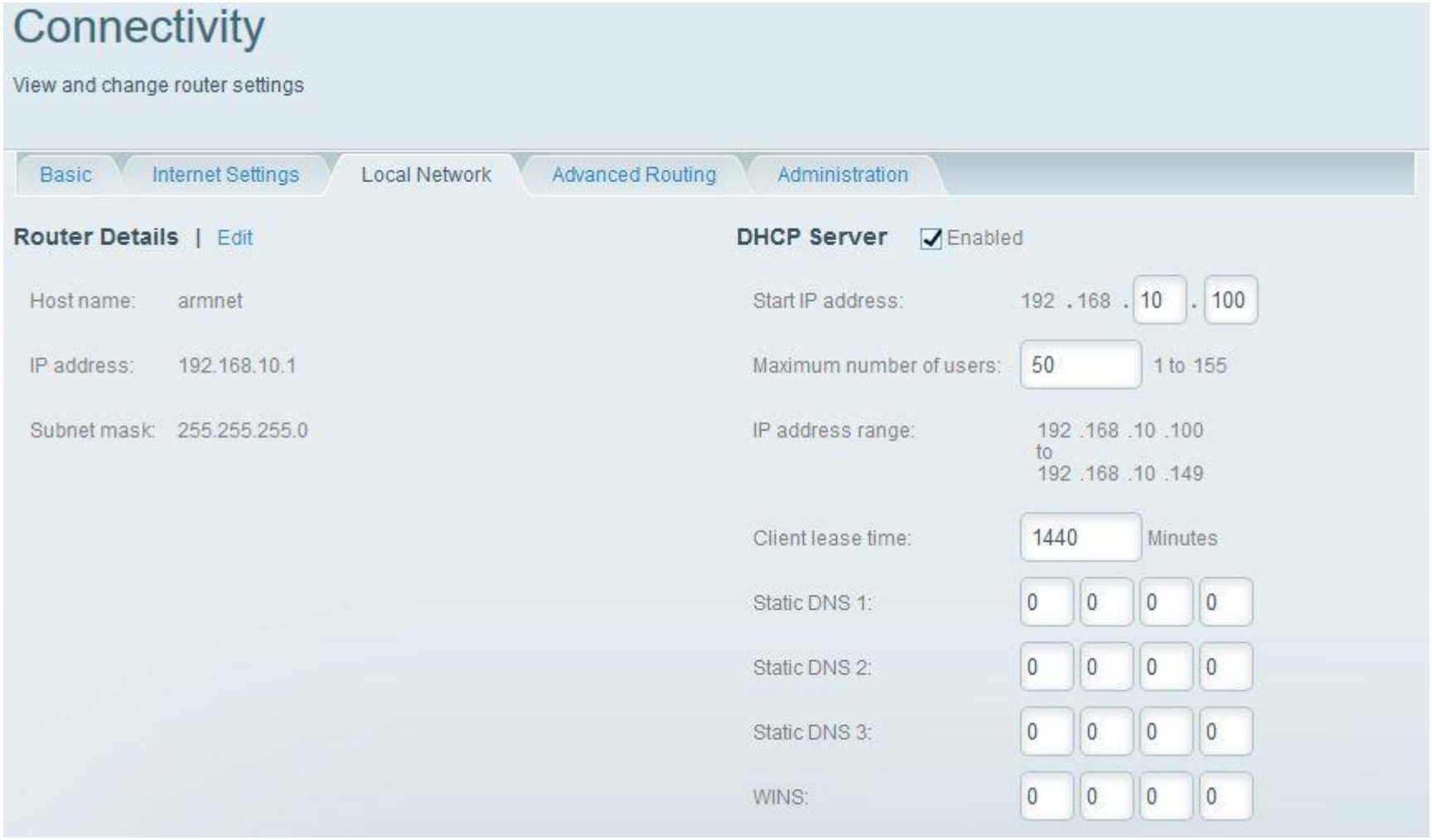
ARM Your Sensors

Lock and Load – Station Mode



ARM Your Sensors

Lock and Load – Station Mode



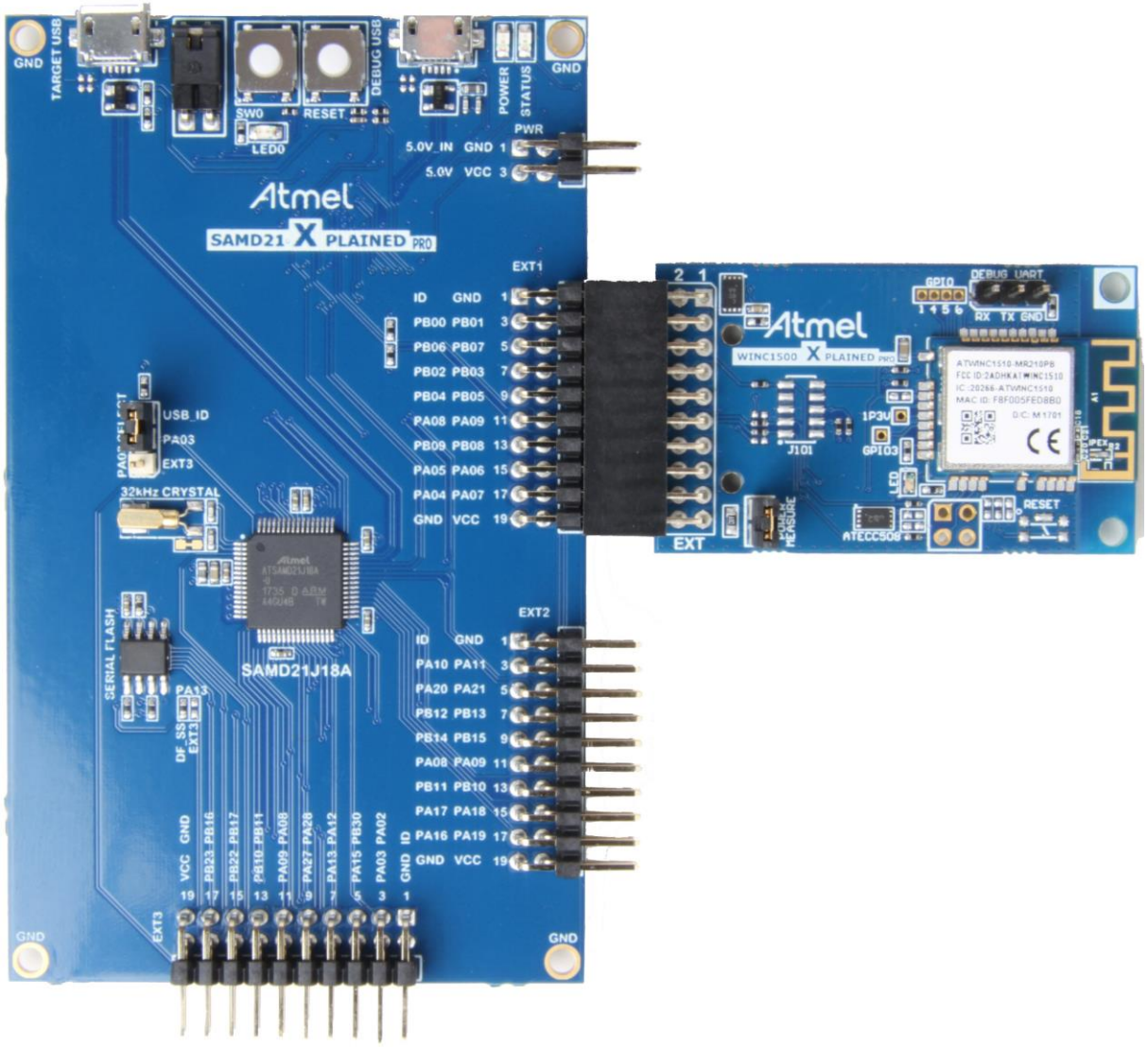
ARM Your Sensors

Lock and Load – Station Mode


```
/** Wi-Fi Settings */
#define MAIN_WLAN_SSID "armwifi"            /* < Destination SSID */
#define MAIN_WLAN_AUTH M2M_WIFI_SEC_WPA_PSK /* < Security manner */
#define MAIN_WLAN_PSK "armwifipass"         /* < Password for Destination SSID */
```

ARM Your Sensors

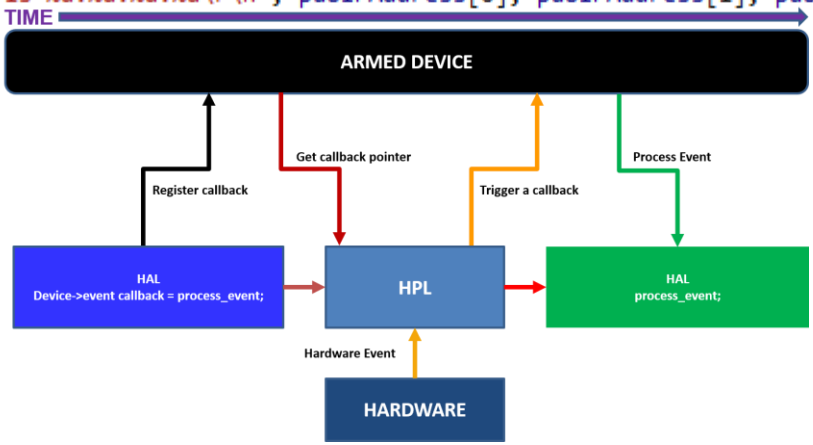
Lock and Load – Station Mode



ARM Your Sensors

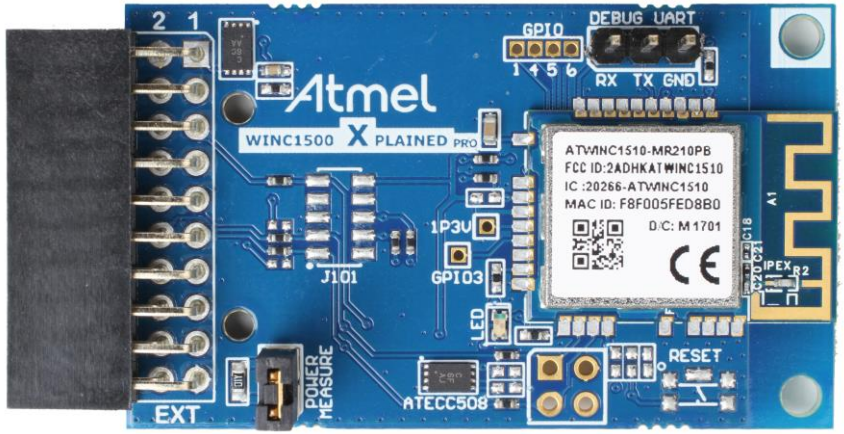
Lock and Load – Station Mode

```
static void wifi_cb(uint8_t u8MsgType, void *pvMsg)
{
    switch (u8MsgType) {
    case M2M_WIFI_RESP_CON_STATE_CHANGED: {
        tstrM2mWifiStateChanged *pstrWifiState = (tstrM2mWifiStateChanged *)pvMsg;
        if (pstrWifiState->u8CurrState == M2M_WIFI_CONNECTED) {
            m2m_wifi_request_dhcp_client();
        } else if (pstrWifiState->u8CurrState == M2M_WIFI_DISCONNECTED) {
            printf("Wi-Fi disconnected\r\n");

            /* Connect to defined AP. */
            m2m_wifi_connect(
                (char *)MAIN_WLAN_SSID, sizeof(MAIN_WLAN_SSID), MAIN_WLAN_AUTH, (void *)MAIN_WLAN_PSK, M2M_WIFI_CH_ALL);
        }

        break;
    }

    case M2M_WIFI_REQ_DHCP_CONF: {
        uint8_t *pu8IPAddress = (uint8_t *)pvMsg;
        printf("Wi-Fi connected\r\n");
        printf("Wi-Fi IP is %u.%u.%u.%u\r\n", pu8IPAddress[0], pu8IPAddress[1], pu8IPAddress[2], pu8IPAddress[3]);
        break;
    }

    default: {
        break;
    }
    }
}
```



ARM Your Sensors

Lock and Load – Station Mode

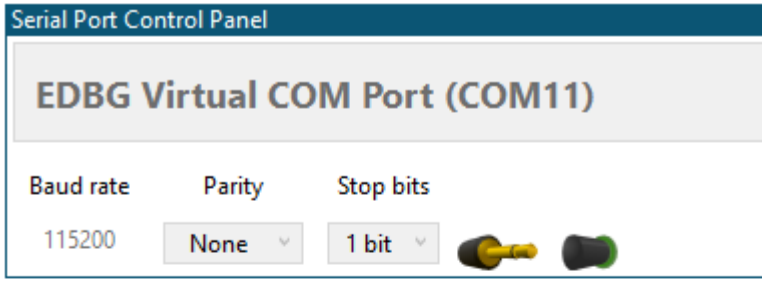


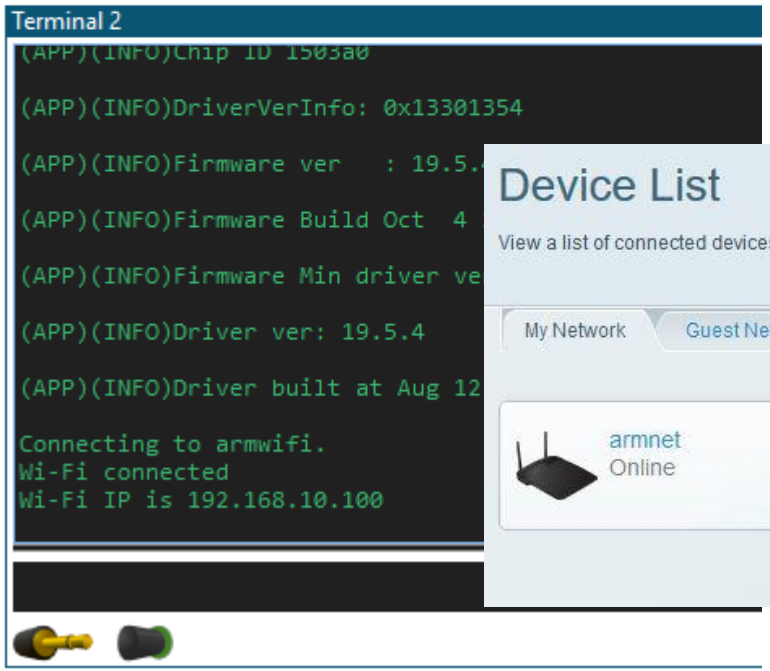

```c
int main(void)
{
    tstrWifiInitParam param;

    atmel_start_init();

    /* Initialize Wi-Fi parameters structure. */
    memset((uint8_t *)&param, 0, sizeof(tstrWifiInitParam));

    /* Initialize Wi-Fi driver with data and status callbacks. */
    param.pfAppWifiCb = wifi_cb;
    wifi_init(&param);

    printf("Connecting to %s.\r\n", (char *)MAIN_WLAN_SSID);
    /* Connect to defined AP. */
    m2m_wifi_connect(
        (char *)MAIN_WLAN_SSID, sizeof(MAIN_WLAN_SSID), MAIN_WLAN_AUTH, (void *)MAIN_WLAN_PSK, M2M_WIFI_CH_ALL);

    while (1) {
        /* Handle pending events from network controller. */
        while (m2m_wifi_handle_events(NULL) != M2M_SUCCESS) {
        }
    }

    return 0;
}
```

ARM Your Sensors

Lock and Load – Station Mode

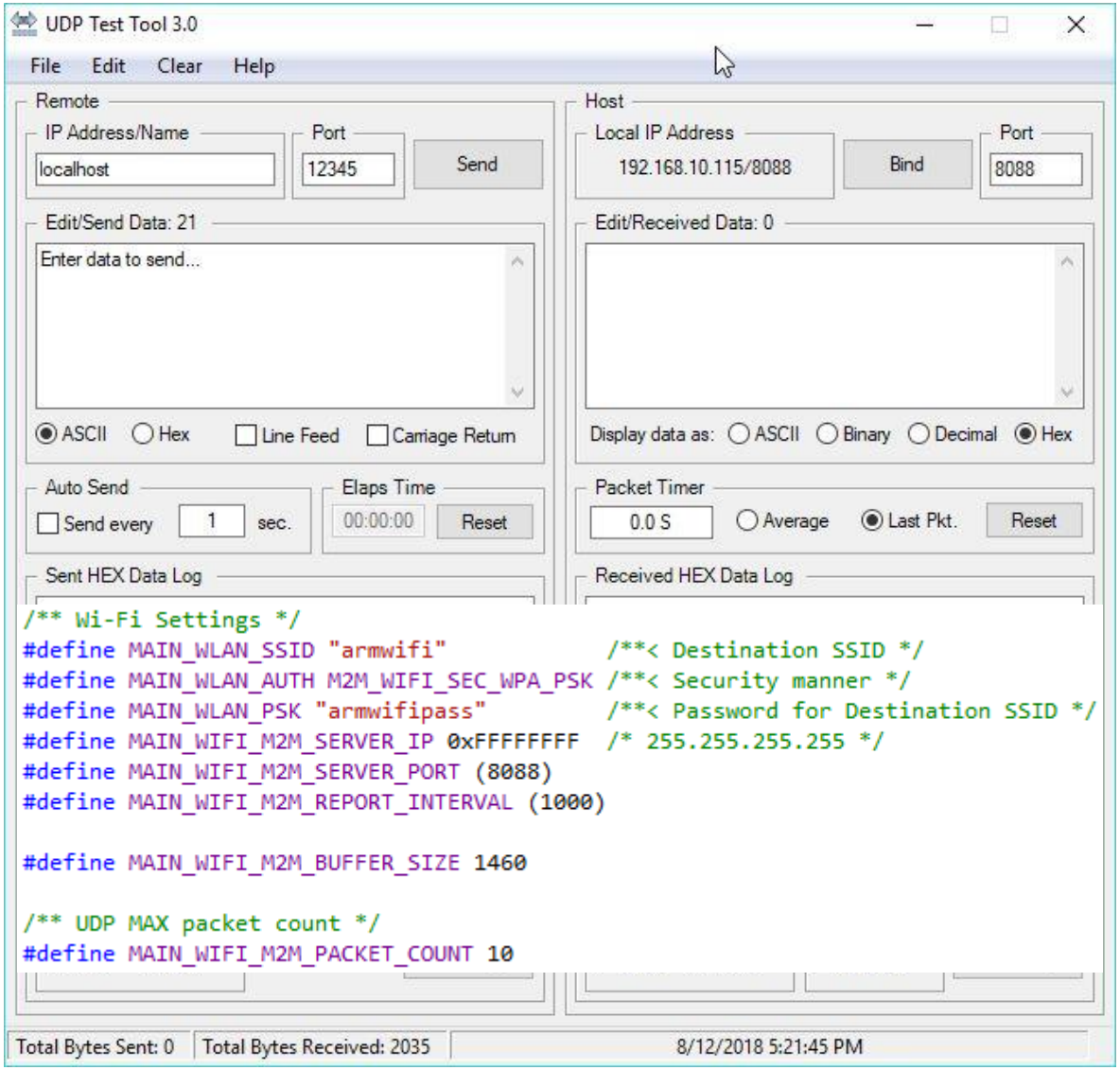
```
case M2M_WIFI_REQ_DHCP_CONF: {
    uint8_t *pu8IPAddress = (uint8_t *)pvMsg;
    printf("Wi-Fi connected\r\n");
    printf("Wi-Fi IP is %u.%u.%u.%u\r\n", pu8IPAddress[0], pu8IPAddress[1], pu8IPAddress[2], pu8IPAddress[3]);
    break;
}
```





ARM Your Sensors

Firing UDP Datagrams



ARM Your Sensors

Firing UDP Datagrams

```
int main(void)
{
    tstrWifiInitParam  param;
    int8_t             ret;
    struct sockaddr_in addr;
    pktData = 0;

    /* Initialize the board. */
    atmel_start_init();

    /* Initialize the BSP. */
    nm_bsp_init();

    /* Initialize socket address structure. */
    addr.sin_family      = AF_INET;
    addr.sin_port        = _htons(MAIN_WIFI_M2M_SERVER_PORT);
    addr.sin_addr.s_addr = _htonl(MAIN_WIFI_M2M_SERVER_IP);

    /* Initialize Wi-Fi parameters structure. */
    memset((uint8_t *)&param, 0, sizeof(tstrWifiInitParam));

    /* Initialize Wi-Fi driver with data and status callbacks. */
    param.pfAppWifiCb = wifi_cb;
    wifi_init(&param);

    /* Initialize socket module */
    socketInit();
```

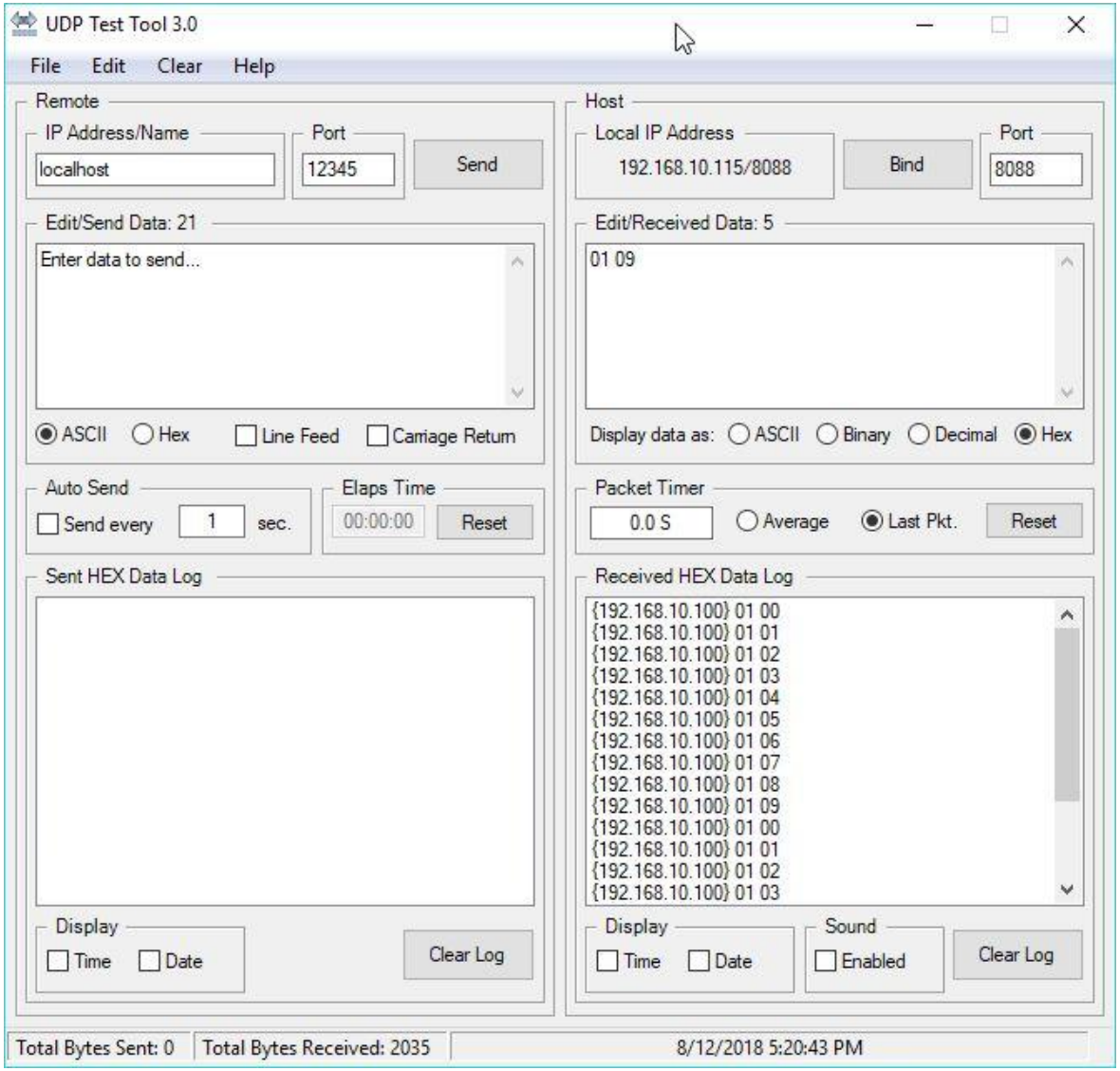

ARM Your Sensors

Firing UDP Datagrams

```
/* Connect to router. */
m2m_wifi_connect(
    (char *)MAIN_WLAN_SSID, sizeof(MAIN_WLAN_SSID), MAIN_WLAN_AUTH, (char *)MAIN_WLAN_PSK, M2M_WIFI_CH_ALL);

while (1) {
    if (packetCnt == MAIN_WIFI_M2M_PACKET_COUNT) {
        close(tx_socket);
        tx_socket = -1;
        break;
    }

    m2m_wifi_handle_events(NULL);

    if (wifi_connected == M2M_WIFI_CONNECTED) {
        /* Create socket for Tx UDP */
        if (tx_socket < 0) {
            if ((tx_socket = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
                printf("main : failed to create TX UDP client socket error!\r\n");
                continue;
            }
        }

        udpPkt[0] = 0x01;
        udpPkt[1] = pktData++;

        ret = sendto(tx_socket,
                     &udpPkt,
                     sizeof(udpPkt),
                     0,
                     (struct sockaddr *)&addr,
                     sizeof(addr));

        if (ret == M2M_SUCCESS) {
            packetCnt += 1;
        }
    }
}
```

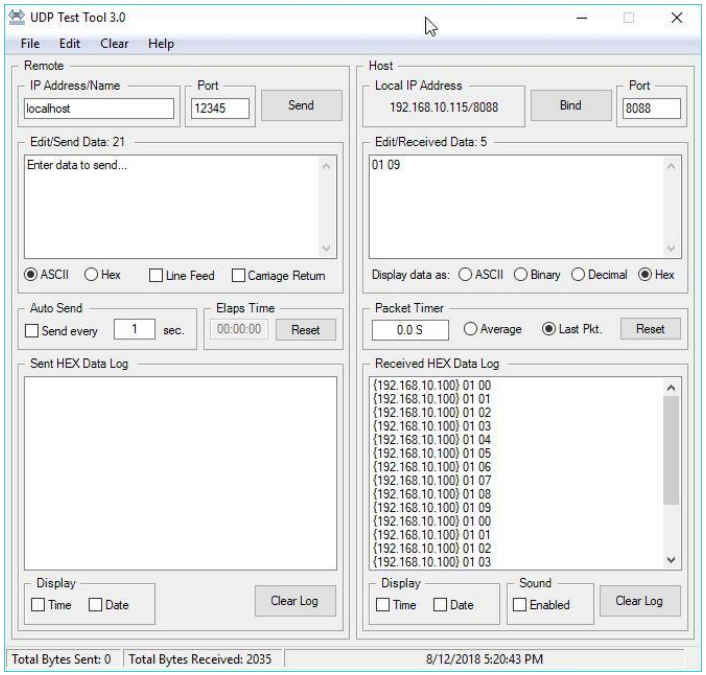
ARM Your Sensors

Firing UDP Datagrams



ARM Your Sensors

Day 1 Summary




```
/* Connect to router. */
m2m_wifi_connect(
    (char *)MAIN_WLAN_SSID, sizeof(MAIN_WLAN_SSID), MAIN_WLAN_AUTH, (char *)MAIN_WLAN_PSK, M2M_WIFI_CH_ALL);

while (1) {
    if (packetCnt == MAIN_WIFI_M2M_PACKET_COUNT) {
        close(tx_socket);
        tx_socket = -1;
        break;
    }

    m2m_wifi_handle_events(NULL);

    if (wifi_connected == M2M_WIFI_CONNECTED) {
        /* Create socket for Tx UDP */
        if (tx_socket < 0) {
            if ((tx_socket = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
                printf("main : failed to create TX UDP client socket error!\r\n");
                continue;
            }
        }

        udpPkt[0] = 0x01;
        udpPkt[1] = pktData++;

        ret = sendto(tx_socket,
                     &udpPkt,
                     sizeof(udpPkt),
                     0,
                     (struct sockaddr *)&addr,
                     sizeof(addr));

        if (ret == M2M_SUCCESS) {
            packetCnt += 1;
        }
    }
}
```