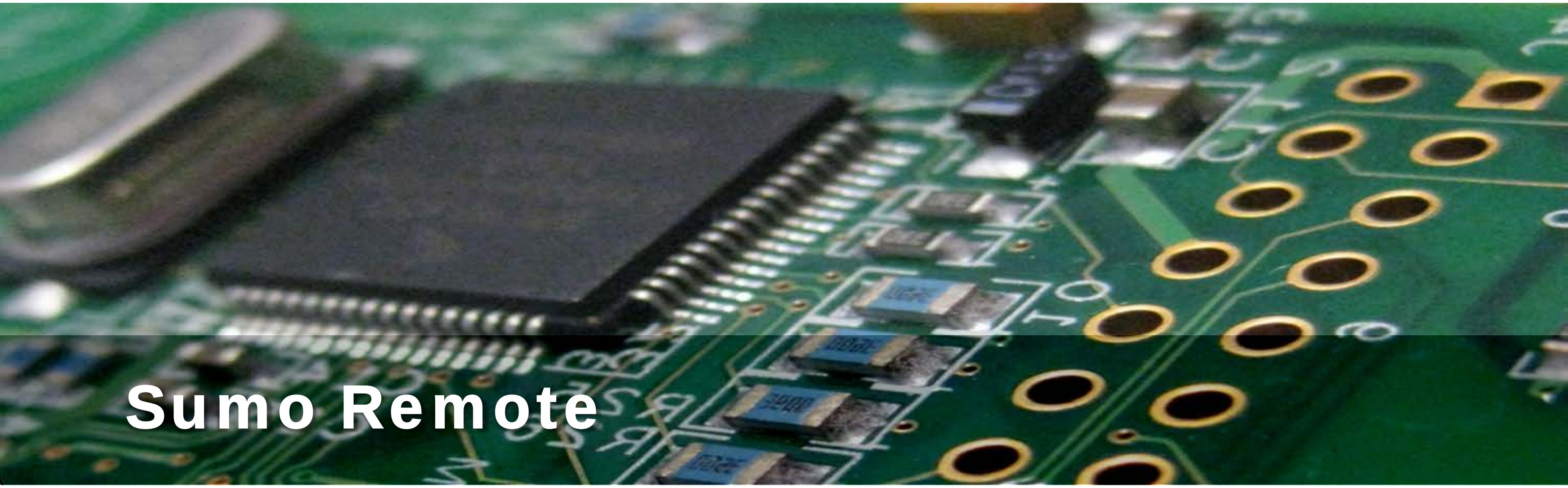
Infos

- Schedule
    - SW14
        - Admission
        - Sumo Remote input, working on bots
    - SW15
        - Mo 29.6.17
            - Q&A, working on bots
        - Tue 30.6.17
            - 0900-????: Sumo competition
            - return lab material
            - Q&A
- MEP
    - Fr. 30.6.17, 1330-1730-45

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur



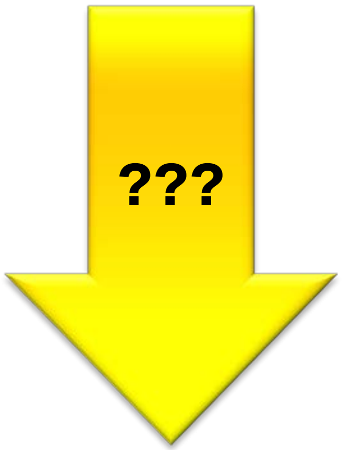
Sumo Remote

“The power is the distance…”

Prof. Erich Styger
erich.styger@hslu.ch
+41 41 349 33 01

Learning Goals

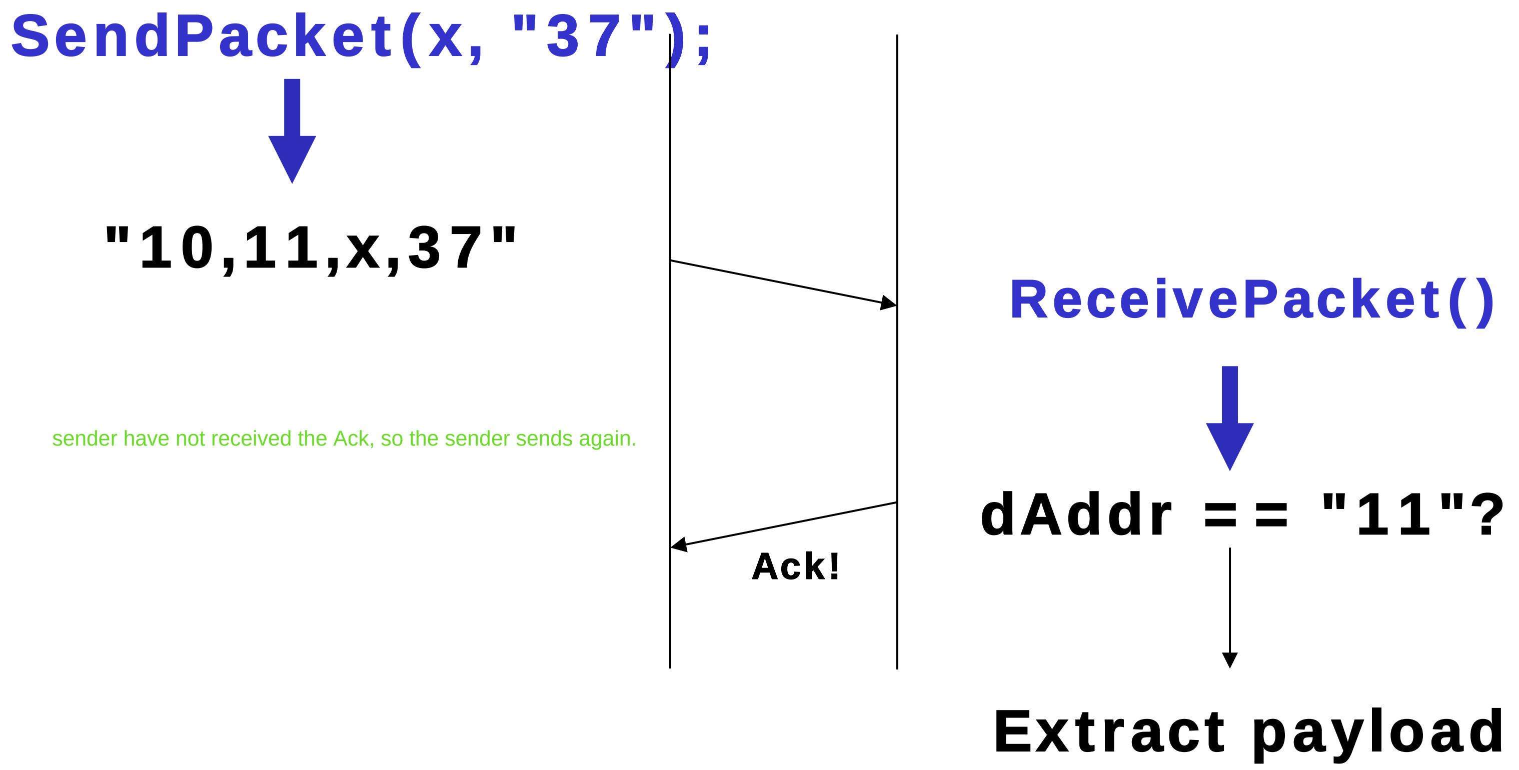
- Communication Protocol
- Radio Message Handlers
- Sending/Receiving Messages
- Application
    - Setting values
    - Getting values
    - Notifications
- Integration with LCD
    - Menu requests and updates



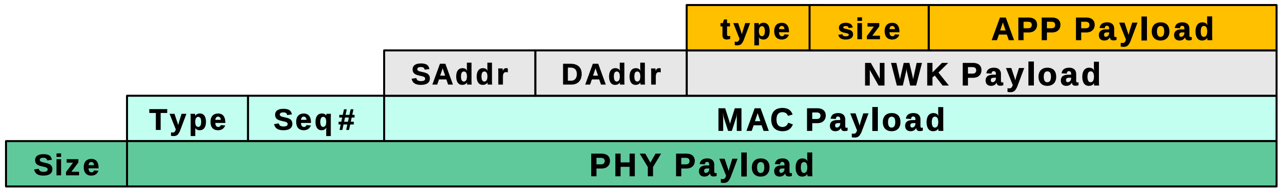
Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Protocol Example

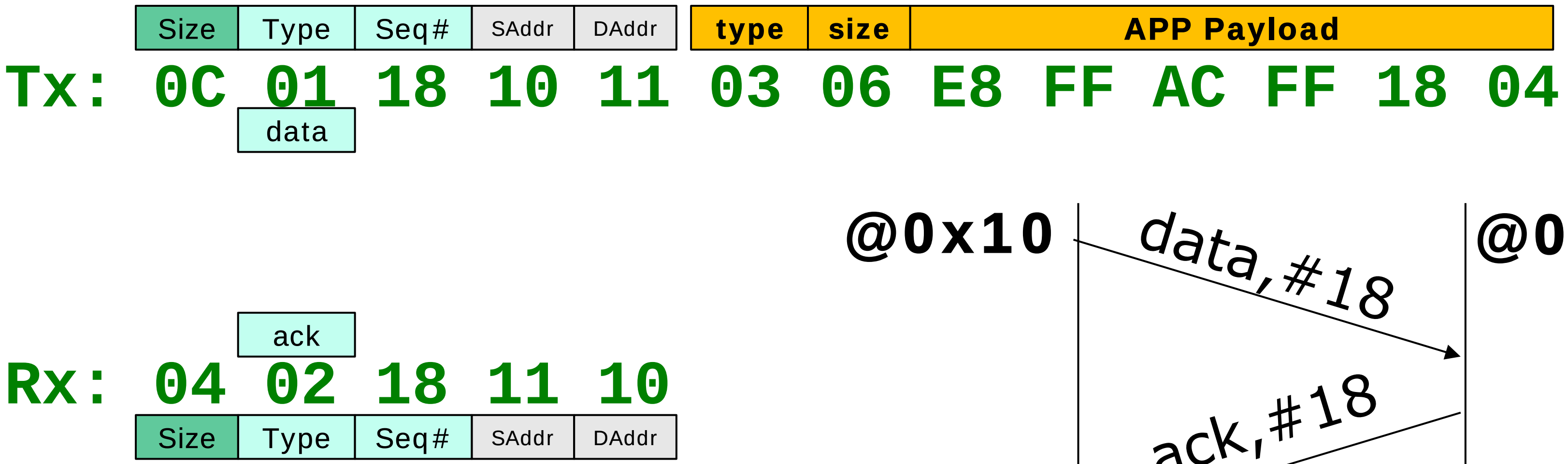
"10->11"	'x'	value
Src, Dst	kind	payload



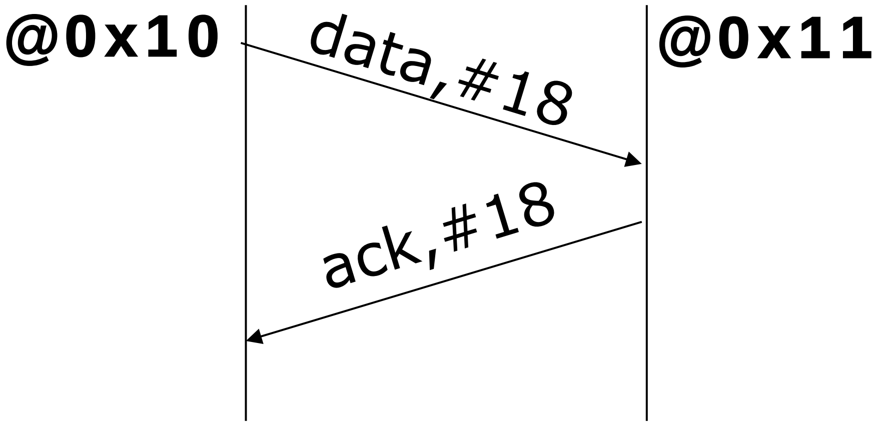
RNet Protocol



```
CMD> radio sniff on
```



```
Tx:  0C 01 18 10 11 03 06 E8 FF AC FF 18 04
Rx:  04 02 18 11 10
```



Application Init and Task

- Initialize stack
- Assign Message Handler
- Assign own node address
- Process Radio State Machine (e.g. in own task)

```
static void RadioTask(void* pvParameters) {
  Init(); /* initialize address */
  appState = RNETA_NONE;
  for(;;) {
    Process(); /* process radio in/out queues */
    vTaskDelay(5/portTICK_PERIOD_MS);
  }
}
```

receive message as soon as possible

```
void RNETA_Init(void) {
  RNET1_Init(); /* initialize stack */
  if (RAPP_SetMessageHandlerTable(handlerTable)!=ERR_OK) { /* assign application message handler */
    for(;;){} /* error */
  }
  if (xTaskCreate(
        RadioTask,  /* pointer to the task */
        "Radio", /* task name for kernel awareness debugging */
        configMINIMAL_STACK_SIZE+100, /* task stack size */
        (void*)NULL, /* optional task startup argument */
```

Application Rx Message Handler

- Table of radio message handlers

```
static const RAPP_MsgHandler handlerTable[] =
{
#if RNET_CONFIG_REMOTE_STDIO
  RSTDIO_HandleStdioRxMessage,
#endif
#if PL_HAS_REMOTE
  REMOTE_HandleRemoteRxMessage,
#endif
  HandleDataRxMessage,
  NULL /* sentinel */
};

static uint8_t HandleDataRxMessage(RAPP_MSG_Type type, uint8_t size, uint8_t *data,
             RNWK_ShortAddrType srcAddr, bool *handled, RPHY_PacketDesc *packet) {
  switch(type) {
    case RAPP_MSG_TYPE_DATA: /* <type><size><data */
      *handled = TRUE;  yes wo know this message, we handle it
      MyVal = *data; /* get data value */
      return ERR_OK;
    default:
      break;
  } /* switch */
  return ERR_OK;
}
```

Sending Data

- Sending Payload Data (Block)
  - Pointer to data, size
  - Message type
  - Destination address

```
uint8_t HandleDataRxMessage(RAPP_MSG_Type type, uint8_t size, uint8_t *data,
            RNWK_ShortAddrType srcAddr, bool *handled, RPHY_PacketDesc *packet) {
  (void)size;
  (void)packet;
  switch(type) {
    case RAPP_MSG_REQUEST_DATA: /* <type><size><data */
      *handled = TRUE;
      accelX = ACCEL_GetX(); /* get accelerometer value */
      return RAPP_SendPayloadDataBlock(&accelX, sizeof(accelX),
          RAPP_MSG_TYPE_RESPONSE, srcAddr, RPHY_PACKET_FLAGS_NONE);
    default:
      break;
  } /* switch */
  return ERR_OK;
}
```

no sending data, just doing in a buffer

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

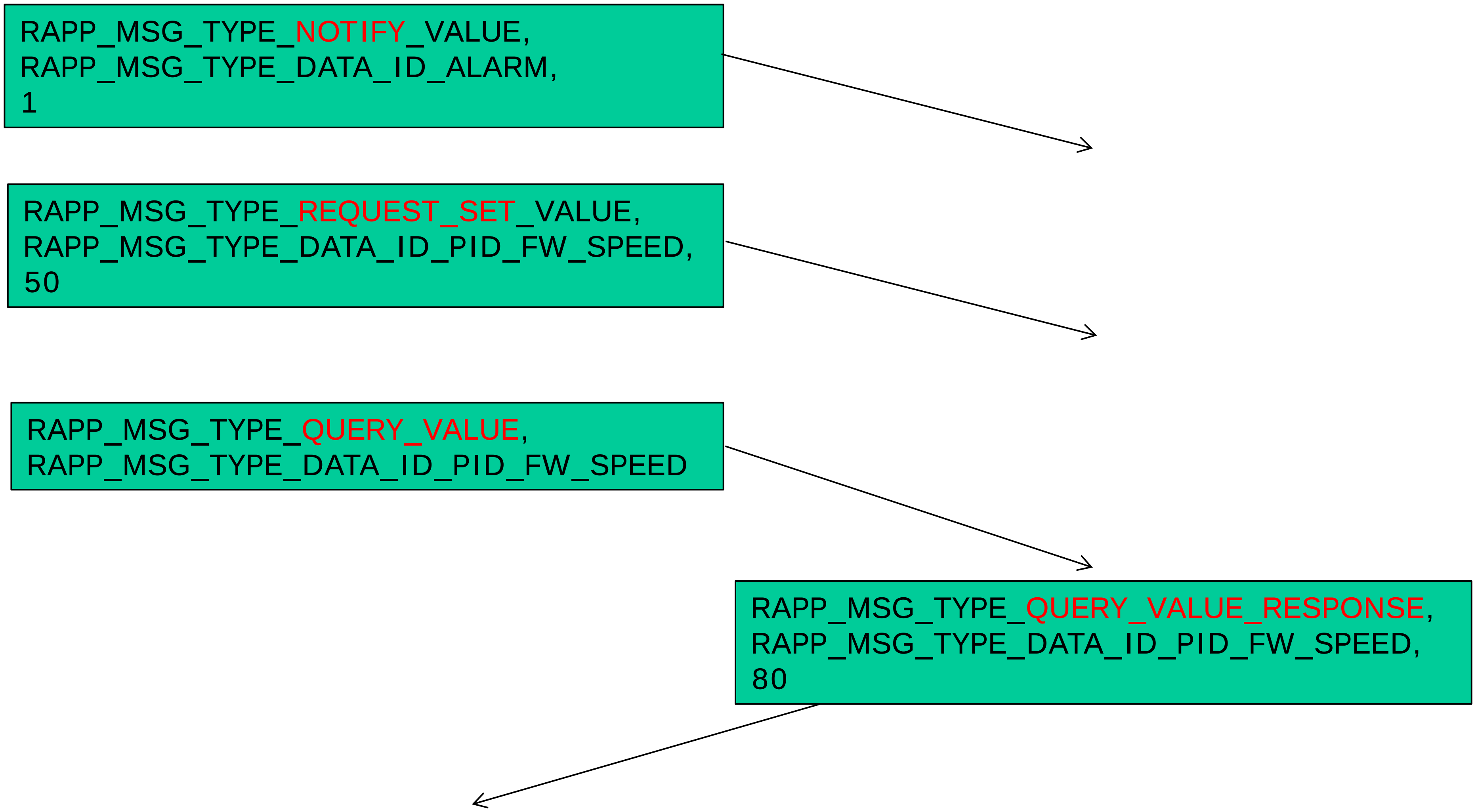
Sending Data as Shell Strings

- Pros
  - Human readable format
  - Re-using infrastructure
  - Extensible
- Cons
  - Data transmission time/packet size

```
case EVNT_SW2_PRESSED:
    LED2_Neg();
    (void)RSTDIO_SendToTxStdio(RSTDIO_QUEUE_TX_IN,
      "buzzer buz 800 400\r\n",
      sizeof("buzzer buz 800 400\r\n")-1);
break;
```

disadvantage: more data for sending

Protocol: Notify, Setter, Getter



RAPP_MSG_TYPE_NOTIFY_VALUE,
RAPP_MSG_TYPE_DATA_ID_ALARM,
1

RAPP_MSG_TYPE_REQUEST_SET_VALUE,
RAPP_MSG_TYPE_DATA_ID_PID_FW_SPEED,
50

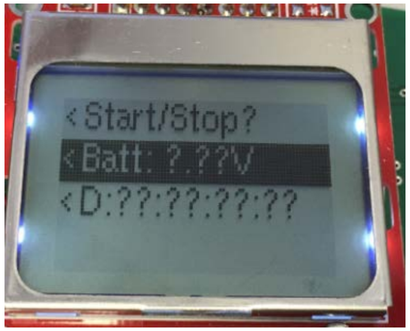
RAPP_MSG_TYPE_QUERY_VALUE,
RAPP_MSG_TYPE_DATA_ID_PID_FW_SPEED

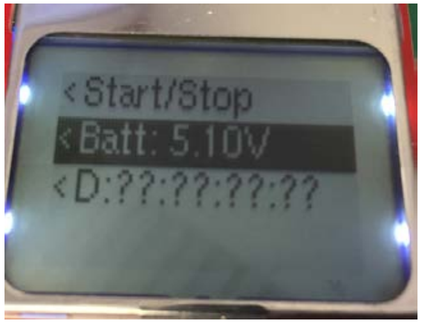
RAPP_MSG_TYPE_QUERY_VALUE_RESPONSE,
RAPP_MSG_TYPE_DATA_ID_PID_FW_SPEED,
80

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Battery Voltage Menu

- **Remote**: LCD to display voltage menu
  - «Batt: 4.57V» or if unknown «Batt: ?.??V»
  - unknown, <ENTER> or <LEFT> on menu
    - Request Battery Voltage from Robot
    - RAPP_MSG_TYPE_QUERY_VALUE for RAPP_MSG_TYPE_DATA_ID_BATTERY_V
- **Robot**: Receives Query, responds with voltage
  - *RAPP_MSG_TYPE_QUERY_VALUE_RESPONSE for* RAPP_MSG_TYPE_DATA_ID_BATTERY_V and voltage
- **Remote**: Receives message
  - Updates data structure
  - Request LCD menu text update





Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Battery Menu: LCD Status

```
struct {
   bool dataValid;
   uint16_t centiV;
   uint8_t str[sizeof("Batt: ?.??V")+1]; /* used to store menu string */
 } battVoltage;
```

```
static LCDMenu_StatusFlags RobotRemoteMenuHandler(const struct LCDMenu_MenuItem_ *item,
                                                  LCDMenu_EventType event, void **dataP) {
  LCDMenu_StatusFlags flags = LCDMENU_STATUS_FLAGS_NONE;

 if (event==LCDMENU_EVENT_GET_TEXT && dataP!=NULL) {
   if (item->id==LCD_MENU_ID_BATTERY_VOLTAGE) {
     UTIL1_strcpy(battVoltage.str, sizeof(battVoltage.str), (uint8_t*)"Batt: ");
     if (battVoltage.dataValid) { /* use valid data */
       UTIL1_strcatNum32sDotValue100(battVoltage.str, sizeof(battVoltage.str), battVoltage.centiV);
     } else { /* request value from robot */
       (void)RNETA_SendIdValuePairMessage(RAPP_MSG_TYPE_QUERY_VALUE,
RAPP_MSG_TYPE_DATA_ID_BATTERY_V, 0, RNWK_ADDR_BROADCAST, RPHY_PACKET_FLAGS_NONE);
       /* use ??? for now until we get the response */
       UTIL1_strcat(battVoltage.str, sizeof(battVoltage.str), (uint8_t*)"?.??");
     }
     UTIL1_strcat(battVoltage.str, sizeof(rbattVoltage.str), (uint8_t*)"V");
     *dataP = battVoltage.str;
     flags |= LCDMENU_STATUS_FLAGS_HANDLED|LCDMENU_STATUS_FLAGS_UPDATE_VIEW;
   }
```

Robot: Request Battery Voltage

```
uint8_t REMOTE_HandleRemoteRxMessage(RAPP_MSG_Type type, uint8_t size, uint8_t
*data,
        RNWK_ShortAddrType srcAddr, bool *handled, RPHY_PacketDesc *packet) {

  switch(type) {
   case RAPP_MSG_TYPE_QUERY_VALUE:
      id = UTIL1_GetValue16LE(data); /* extract 16bit ID (little endian) */
      if (id==RAPP_MSG_TYPE_DATA_ID_BATTERY_V) {
         uint16_t centiV;

        if (BATT_MeasureBatteryVoltage(&centiV)!=ERR_OK) {
          centiV = 0; /* error case */
        }
        RNETA_SendIdValuePairMessage(RAPP_MSG_TYPE_QUERY_VALUE_RESPONSE, id,
                    centiV, srcAddr, RPHY_PACKET_FLAGS_NONE);
        *handled = TRUE;
        beep =  TRUE;
      }
     ....
```

LCD: Receiving Battery Voltage

```
uint8_t LCD_HandleRemoteRxMessage(RAPP_MSG_Type type, uint8_t size,
uint8_t *data,
    RNWK_ShortAddrType srcAddr, bool *handled, RPHY_PacketDesc *packet) {


 switch(type) {
   case RAPP_MSG_TYPE_QUERY_VALUE_RESPONSE: /* receive data value */
     msgID = UTIL1_GetValue16LE(&data[0]); /* ID in little endian format */
     if (msgID==RAPP_MSG_TYPE_DATA_ID_BATTERY_V){
       *handled = TRUE;
       msgValue = UTIL1_GetValue32LE(&data[2]);
       remoteValues.battVoltage.centiV = msgValue;
       requestLCDUpdate = TRUE;
       remoteValues.battVoltage.dataValid = TRUE;
     }
     break;
```

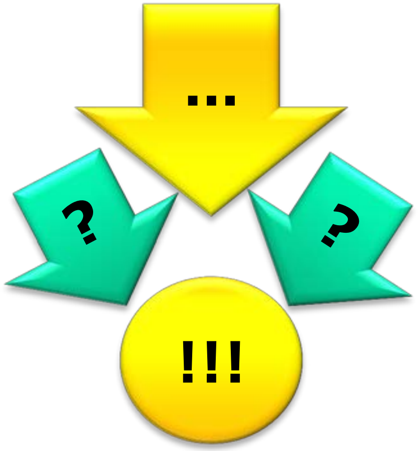
we need to extract the ID

difference between little endian and big endian

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Summary

- Simple peer to peer communication
  - PAIND and/or addressing
  - Data format
    - type, seq#, saddr, daddr, application payload
- Messages
  - Setter, Getter, Notifications



Lab: Remote

- Integrate
  - Menu Handler
  - Message Handler
- Send Messages
  - Notification
  - Setter
  - Getter
- Define your own format/messages