Have a look at the following pages in the device Datasheet and Technical Reference Manual, and then change the functions in the files of hal.c, hal.h and user.h according to the hardware board.

```
//SPRUI33B, P90, Table 3-3. PIE Channel Mapping
//SPRUI33B, P1311-1313, Figure 12-1,12-2,12-3, Analog Subsystem Block Diagram
//SPRUI33B, P1314, Figure 12-4. Analog Group Connections
//SPRUI33B, P1720, Figure 18-48. Digital-Compare Submodule High-Level Block Diagram
//SPRUI33B, P1721, Figure 18-49. GPIO MUX-to-Trip Input Connectivity

//SPRS945D, P40-41, Table 4-6. GPIO Muxed Pins
//SPRS945D, P94-96, Figure 5-28, 5-29, 5-30. Analog Subsystem Block Diagram
//SPRS945D, P97, Figure 5-31. Analog Group Connections
//SPRS945D, P98-99, Table 5-39. Analog Pins and Internal Connections
//SPRS945D, P121, Figure 5-48. CMPSS Connectivity
```

1. Change HAL_init() in the file of hal.c

```
HAL_Handle HAL_init(void *pMemory,const size_t numBytes)
{

    // initialize PWM handles for Motor
    obj->pwmHandle[0] = EPWM1_BASE;        //!< the PWM handle
    obj->pwmHandle[1] = EPWM2_BASE;        //!< the PWM handle,
    obj->pwmHandle[2] = EPWM3_BASE;        //!< the PWM handle,

    // initialize PGA handle
    obj->pgaHandle[0] = PGA1_BASE;         //!< the PGA handle
    obj->pgaHandle[1] = PGA3_BASE;         //!< the PGA handle
    obj->pgaHandle[2] = PGA5_BASE;         //!< the PGA handle

    // initialize CMPSS handle
    obj->cmpssHandle[0] = CMPSS1_BASE;     //!< the CMPSS handle
    obj->cmpssHandle[1] = CMPSS3_BASE;     //!< the CMPSS handle
    obj->cmpssHandle[2] = CMPSS5_BASE;     //!< the CMPSS handle


} // end of HAL_init() function
```

2. Change HAL_setupADCs() in the file of hal.c

```
void HAL_setupADCs(HAL_Handle handle)
{


  // configure the interrupt sources
  // B0->RB1->INT_ADCB1
  ADC_setInterruptSource(obj->adcHandle[1], ADC_INT_NUMBER1, ADC_SOC_NUMBER1);

  // configure the SOCs for hvkit_rev1p1
  // IU-FB - C2->RC0
  ADC_setupSOC(obj->adcHandle[2], ADC_SOC_NUMBER0, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN2, HAL_ADC_SAMPLE_WINDOW);

  // IV-FB - A1->RA0
  ADC_setupSOC(obj->adcHandle[0], ADC_SOC_NUMBER0, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN1, HAL_ADC_SAMPLE_WINDOW);
  // IW-FB - B15->RB0
  ADC_setupSOC(obj->adcHandle[1], ADC_SOC_NUMBER0, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN15, HAL_ADC_SAMPLE_WINDOW);

  // Vu-FB - C8->RC1
  ADC_setupSOC(obj->adcHandle[2], ADC_SOC_NUMBER1, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN8, HAL_ADC_SAMPLE_WINDOW);

  // Vv-FB - A10->RA1
  ADC_setupSOC(obj->adcHandle[0], ADC_SOC_NUMBER1, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN10, HAL_ADC_SAMPLE_WINDOW);

  // Vw-FB - B0->RB1
  ADC_setupSOC(obj->adcHandle[1], ADC_SOC_NUMBER1, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN0, HAL_ADC_SAMPLE_WINDOW);

  // VDCBUS - A6->RA2
  ADC_setupSOC(obj->adcHandle[0], ADC_SOC_NUMBER2, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN6, HAL_ADC_SAMPLE_WINDOW);

  // VBAT   - A9->RA3
  ADC_setupSOC(obj->adcHandle[0], ADC_SOC_NUMBER3, ADC_TRIGGER_EPWM1_SOCA,
               ADC_CH_ADCIN9, HAL_ADC_SAMPLE_WINDOW);

  return;
} // end of HAL_setupADCs() function
```

3. Change enableADCInts() in the file of hal.c

```
void HAL_enableADCInts(HAL_Handle handle)
{


    // enable the PIE interrupts associated with the ADC interrupts
    Interrupt_enable(INT_ADCB1);

    // enable the ADC interrupts
    ADC_enableInterrupt(obj->adcHandle[1], ADC_INT_NUMBER1);

    // enable the cpu interrupt for ADC interrupts
    Interrupt_enableInCPU(INTERRUPT_CPU_INT1);

    return;
} // end of HAL_enableADCInts() function
```

4. In HAL_initIntVectorTable()of hal.h

```
static inline void HAL_initIntVectorTable(HAL_Handle handle)
{

    Interrupt_register(INT_ADCB1, &mainISR);

    return;
} // end of HAL_initIntVectorTable() function
```

5. Change HAL_ackADCInt() in the file of hal.h

```
static inline void
HAL_ackADCInt(HAL_Handle handle,const ADC_IntNumber adcIntNum)
{


  // clear the ADC interrupt flag
  // RB1
  ADC_clearInterruptStatus(obj->adcHandle[1], adcIntNum);

  // Acknowledge interrupt from PIE group 1
  Interrupt_clearACKGroup(INTERRUPT_ACK_GROUP1);

} // end of HAL_ackADCInt() function
```

6. Change HAL_readADCDataWithOffsets() in the file of hal.h

```
static inline void
HAL_readADCDataWithOffsets(HAL_Handle handle,HAL_ADCData_t *pADCData)
{

  float32_t current_sf = HAL_getCurrentScaleFactor(handle); // change the sign accordingly
  float32_t voltage_sf = HAL_getVoltageScaleFactor(handle);

  // convert phase U current        C2->RC0
  value = (float32_t)ADC_readResult(obj->adcResult[2], ADC_SOC_NUMBER0);
  pADCData->I_A.value[0] = value * current_sf;

  // convert phase V current        A1->RA0
  value = (float32_t)ADC_readResult(obj->adcResult[0], ADC_SOC_NUMBER0);
  pADCData->I_A.value[1] = value * current_sf;

  // convert phase W current        B15->RB0
  value = (float32_t)ADC_readResult(obj->adcResult[1], ADC_SOC_NUMBER0);
  pADCData->I_A.value[2] = value * current_sf;

  // convert phase U voltage        C8->RC1
  value = (float32_t)ADC_readResult(obj->adcResult[2], ADC_SOC_NUMBER1);
  pADCData->V_V.value[0] = value * voltage_sf;

  // convert phase V voltage        A10->RA1
  value = (float32_t)ADC_readResult(obj->adcResult[0], ADC_SOC_NUMBER1);
  pADCData->V_V.value[1] = value * voltage_sf;

  // convert phase W voltage        B0->RB1
  value = (float32_t)ADC_readResult(obj->adcResult[1], ADC_SOC_NUMBER1);
  pADCData->V_V.value[2] = value * voltage_sf;

  // convert dcBus voltage          A6->RA2
  value = (float32_t)ADC_readResult(obj->adcResult[0], ADC_SOC_NUMBER2);
  pADCData->dcBus_V = value * voltage_sf;

} // end of HAL_readADCDataWithOffsets() function
```

NOTE: Change the sign of the current coefficient according to the current sensing circuit.

```
        current_sf =  HAL_getCurrentScaleFactor(handle);  // Positive feedback
     Or current_sf = -HAL_getCurrentScaleFactor(handle);  // Negative feedback
```

7. Change **HAL_setupCMPSSs()** in file of **hal.c**

```
void HAL_setupCMPSSs(HAL_Handle handle)
{

    ASysCtl_selectCMPHPMux(ASYSCTL_CMPHPMUX_SELECT_1, 4);
    ASysCtl_selectCMPLPMux(ASYSCTL_CMPLPMUX_SELECT_1, 4);

    ASysCtl_selectCMPHPMux(ASYSCTL_CMPHPMUX_SELECT_3, 4);
    ASysCtl_selectCMPLPMux(ASYSCTL_CMPLPMUX_SELECT_3, 4);

    ASysCtl_selectCMPHPMux(ASYSCTL_CMPHPMUX_SELECT_5, 4);
    ASysCtl_selectCMPLPMux(ASYSCTL_CMPLPMUX_SELECT_5, 4);

    // Configure TRIP9 to be CTRIP1H and CTRIP1L using the ePWM X-BAR
    XBAR_setEPWMMuxConfig(XBAR_TRIP9, XBAR_EPWM_MUX08_CMPSS5_CTRIPH_OR_L);
    XBAR_enableEPWMMux(XBAR_TRIP9, XBAR_MUX08);

    // Configure TRIP7 to be CTRIP1H and CTRIP1L using the ePWM X-BAR
    XBAR_setEPWMMuxConfig(XBAR_TRIP7, XBAR_EPWM_MUX00_CMPSS1_CTRIPH_OR_L);
    XBAR_enableEPWMMux(XBAR_TRIP7, XBAR_MUX00);

    // Configure TRIP8 to be CTRIP1H and CTRIP1L using the ePWM X-BAR
    XBAR_setEPWMMuxConfig(XBAR_TRIP8, XBAR_EPWM_MUX04_CMPSS3_CTRIPH_OR_L);
    XBAR_enableEPWMMux(XBAR_TRIP8, XBAR_MUX04);

} // end of HAL_setupCMPSSs() function
```

8. Change **HAL_setupFaults()** in the file of **hal.c**

```
void HAL_setupFaults(HAL_Handle handle)
{
    HAL_Obj *obj = (HAL_Obj *)handle;
    uint16_t cnt;

    // configure the input x bar for TZ2 to GPIO, where Over Current is connected
    XBAR_setInputPin(XBAR_INPUT2, HAL_PM_nFAULT_GPIO);
        XBAR_lockInput(XBAR_INPUT2);

} // end of HAL_setupFaults() function
```

9. Change current and voltage scale, analog filter pole frequency in file of **user.h**

```
//! \brief Defines the nominal DC bus voltage, V
//!
#define USER_NOMINAL_DC_BUS_VOLTAGE_V        ((float32_t)(400.0))
```

```c
//! \brief Defines the maximum voltage at the AD converter
//!
#define USER_ADC_FULL_SCALE_VOLTAGE_V         ((float32_t)(409.6)

//! \brief Defines the maximum current at the AD converter
//!
#define USER_ADC_FULL_SCALE_CURRENT_A         ((float32_t)(19.89))

//! \brief Defines the analog voltage filter pole location, Hz
//!
#define USER_VOLTAGE_FILTER_POLE_Hz           ((float32_t)(372.5))
```