TUBii Server API Description

Eric Marzec

December 12, 2015

Described here is the set of commands I think would be sufficient and convenient for the server runing on TUBii to implement.

-
  ```
  SetGTDelays(uint8 LO_Delay, uint8 DGT_Delay)
  {
  muxer 3
  muxenable 1
  loadshift LO_Delay
  loadshift DGT_Delay
  muxenable 0
  self.LO_Delay = LO_Delay self.DGT_Delay = DGT_Delay
  return ErrorDidHappen?
  }
  ```

-
  ```
  GetLODelay() {
  return self.LO_Delay
  }
  ```

-
  ```
  GetDGTDelay() {
  return self.DGT_Delay
  }
  ```

-
  ```
  SetCaenWords( uint8 GainPathWord, uint8 ChannelSelectWord)
  {
  muxer 1
  muxenable 1
  loadshift GainPathWord
  loadshift ChannelSelectWord
  muxennable 0
  dataready 6
  ```

```
dataready 4

self.GainPathWord = GainPathWord
self.ChannelSelectWord = ChannelSelectWord

return ErrorDidHappen?
}
```

-
  ```
  GetCAENGainPathWord()
  {
  return self.GainPathWord
  }
  ```

-
  ```
  GetCAENChannelSelectWord()
  {
  return self.ChannelSelectWord
  }
  ```

-
  ```
  SetControlReg(uint8 ControlRegWord)
  {
  muxer 0
  muxenable 1
  loadshift ControlRegWord
  muxenable 0
  dataready 5
  dataready 4

  self.ControlRegWord = ControlRegWord
  (Perhaps here is where we would like to use the ReadShift command
  to confirm that things actually worked)
  return ErrorDidHappen?
  }
  ```

-
  ```
  GetControlReg()
  {
  (Maybe we want to use the readshift command here as well)
  return self.ControlRegWord
  ```

}

- SetDACThreshold(uint16 DACWord) {
  muxer 2
  muxenable 1
  muxenable 0
  (The DAC word is 16 bits (12 actually) but loadshift currently writes 8 at a time. You'll have to figure out how you wanna deal with this. Hopefully that won't be too difficult)

  loadshift DACWord1
  loadshift DACWord2
  dataready 0
  dataready 4

  self.DACWord = DACWord

  return ErrorDidHappen?
  }

- GetDACThreshold()
  {
  return self.DACWord
  }

- SetAllowableClockMisses(uint8 NMisses)
  {
  (I've actually only ever changed this on the TUBii once. So I don't actually know the commands. I'll have to figure this one out later)

  self.AllowableMisses = self.AllowableMisses

  return ErrorDidHappen? }

- smelliePulser(rate, width,nPulses)
  {
  (Whatever this function already does)

```
self.smellieRate = rate
self.smellieWidth = width
self.smellieNPulses = nPulses

return ErrorDidHappen?
}
```

-
  ```
  GetSmellieRate()
  {
  return self.smellieRate
  }
  ```

-
  ```
  GetSmellieWidth()
  {
  return self.smellieWidth
  }
  ```

-
  ```
  GetSmellieNPulses()
  {
  return self.smellieNPulses
  }
  ```

-
  ```
  smellieDelay(length)
  {
  (Whatever this function already does)
  self.smellieDelayLength = Length
  }
  ```

-
  ```
  GetSmellieDelay()
  {
  return self.smellieDelayLength }
  ```

- Repeat all the smellie functions for tellie

- Repeat all the smellie functions for Generic

- clockReset()
  {
  sets the clock reset pin high than low
  This is somewhat different than the current version of clockReset which requires an arguement)

  return ErrorDidHappen? }

- clockStatus()
  {
  (same as what it alredy does except returns value not in datastream)
  }

- countMask(countMask)
  {
  (same as what it alredy does)
  self.countMask = countMask
  return ErrorDidHappen?
  }

- GetCountMask()
  {
  return self.countMask
  }

- Same functions that count has except for the speaker

- Getters/Setters for trigger mask

- Something to do with setting combo/burst/prescale?
  You probably know better about how necessary getters/setters for that are