|  |  |  |
| --- | --- | --- |
| **Address** | **Element** | **Description** |
| **0** | **partNum** | Device ID: uint16\_t  This ID is used to identify the ADPD4100 sensor device type  and do specific tuning.  0x00C0 – ADPD4000 device – on DVT1 watch  0x01C2 – ADPD4100 device - on DVT2/3 watch |
| **1** | **targetSlots** | Slot selection: uint16\_t  Determine the slot from the 12 slots available on ADPD4000/ADPD4100 |
| **2** | **targetChs** | Target Channel(s) selection and shift: uint8\_t  The lower nibble chooses the input channel value to HRM from the slot (selected by targetSlots) while upper nibble chooses the shift value of channel signal in mode 4.  Bit[3:0]: channel mode  Bit[7:4]: shift (for mode 4)  1 - Channel 1 (default)  2 - Channel 2  3 – Channel1 will be fed as PD1+PD2  4 – Channel1 and Channel2 shifted by value in [7:4] and sum.  Eg: - 0x24 🡪( Channel1>>2+Channel2>>2),  0x14 🡪( Channel1>>1+Channel2>>1), |
| **3** | **deviceMode** | Not used |
| **4** | **featureSelect** | Enable pre-process features: uint16\_t  Used to enable/disable some of the pre-process states of the ADPD State Machine. This should not be used unless the user is fully aware of what they are doing.  Bit is 1: feature enabled; 0: feature disabled  Bit 0 🡪 Detect On  Bit 1 🡪 Detect Off  Bit 4 🡪 Not used  Bit 6 🡪 Not used  Bit 8 🡪 Dynamic AGC  Bit 9 🡪 Static AGC  Bit 12 🡪 HRM algorithm |
| **5** | **drTime** | Data Rate Time: uint16\_t  Sets the time used to determine the data rate adjustment factor.  To account for the fact that clock calibration does not align exactly on the rate needed. The data rate is determined during the running of the algorithm; this factor is then used to adjust the heart rate value determined by the algorithm.  The input is a time in milliseconds. The default is 0ms. |
| **6** | **DutyCycle** | Not used (uint32\_t) |
| **7** | **hrmInputRate** | Used to check the sampling rate to be used for the algorithm. ADI HRM Algorithm used in this package will support only 50Hz: uint16\_t  To support a higher sample rate of ADPD and ADXL, both data should be decimated to 50Hz. |
| **8** | **syncMode** | Not used |
| **9** | **proximityRate** | Not used |
| **10** | **proximityTimeout** | Not used |
| **11** | **proximityOnLevel** | Not used |
| **12** | **Res32\_1** | Reserved(uint32\_t) |
| **13** | **Res16\_1** | Reserved(uint16\_t) |
| **14** | **Res8\_1** | Reserved(uint8\_t) |
| **15** | **Res16\_2** | Reserved(uint16\_t) |
| **16** | **Res16\_3** | Reserved(uint16\_t) |
| **17** | **Res32\_2** | Reserved(uint32\_t) |
| **18** | **initialLedPulse** | Initial Pulse for Static AGC in PPG application: uint16\_t  When static AGC is not enabled, this will be the pulse for the session. |
| **19** | **Res8\_2** | Reserved(uint8\_t) |
| **20** | **Res8\_3** | Reserved(uint8\_t) |
| **21** | **Res32\_3** | Reserved(uint32\_t) |
| **22** | **rmssdSampleWindow** | RMSSD calculation window size: uint16\_t  This is used during RMSSD calculation of HRV data. |
| **23** | **Res8\_4** | Reserved (uint8\_t) |
| **24** | **Res16\_4** | Reserved(uint16\_t) |
| **25** | **Res16\_5** | Reserved(uint16\_t) |
| **26** | **maxSamplingRate** | Maximum sampling rate used for Dynamic AGC: uint16\_t |
| **27** | **targetDcPercent** | Target DC level in percentage: uint8\_t  Used in dynamic AGC to set the target current percentage during AFE saturation. It is also used for checking DC level during pulse adjust |
| **28** | **maxLedCurrent** | Set Maximum Current for dynamic AGC: uint16\_t |
| **29** | **maxPulseNum** | Set Maximum LED pulses dynamic AGC: uint8\_t |
| **30** | **satAdjustPercentForStaticAgc** | Saturation adjust in percentage: uint8\_t  Used in Static AGC to set the target current percentage during AFE saturation.  (It is also used by ppg application when static AGC is disabled) |
| **31** | **Res8\_5** | Reserved(uint8\_t) |
| **32** | **InitialCurrentTiaGain** | LED current & TIA gain when static AGC is disabled: uint16\_t  Eg:- 0000BF24 -> current(bit[15:8]) and Initial TIA Gain channel2 [5:3] channel1[2:0])  where,  current is the actual LED current  Initial TIA Gain setting is  000: 200 kΩ.  001: 100 kΩ.  010: 50 kΩ.  011: 25 kΩ.  100: 12.5 kΩ |
| **33** | **motionThreshold** | Low Motion activity threshold(For dynamic AGC ): uint32\_t |
| **34** | **motionCheckPeriod** | Upper = check period, Lower = rest time(dynamic AGC): uint32\_t  Default set to Upper=2min and Lower=3sec |
| **35** | **motionThresholdHigh** | High motion activity threshold (dynamic AGC): uint32\_t |
| **36** | **motionCheckPeriodHigh** | Low & High motion activity period check (dynamic AGC): uint32\_t  Default set to 6sec for low and high activity |
| **37** | **Res8\_6** | Reserved (uint8\_t) |
| **38** | **Res16\_6** | Reserved(uint16\_t) |
| **39** | **Res16\_7** | Reserved(uint16\_t) |
| **40** | **sqiLowPowerThreshold** | SQI low power threshold: uint16\_t  If SQI is greater than this threshold, then dynamic AGC will decrease the power. This means the signal is good.  Default: (0.25\*1024) |
| **41** | **sqiHighPowerThreshold** | SQI high power threshold: uint16\_t  If SQI is less than or equal to this threshold, then dynamic AGC will increase the power. This means the signal is poor.  Default: (0.15\*1024) |