MPU6050/MPU6050.h

```
00001 // I2Cdev library collection - MPU6050 I2C device class
00002 // Based on InvenSense MPU-6050 register map document rev. 2.0, 5/19/2011 (RM-MPU-6000A-00)
00003 // 10/3/2011 by Jeff Rowberg <jeff@rowberg.net>
00004 // Updates should (hopefully) always be available at https://github.com/jrowberg/i2cdevlib
00005 //
00006 // Changelog:
00007 //     ... - ongoing debug release
00008
00009 // NOTE: THIS IS ONLY A PARIAL RELEASE. THIS DEVICE CLASS IS CURRENTLY UNDERGOING ACTIVE
00010 // DEVELOPMENT AND IS STILL MISSING SOME IMPORTANT FEATURES. PLEASE KEEP THIS IN MIND IF
00011 // YOU DECIDE TO USE THIS PARTICULAR CODE FOR ANYTHING.
00012
00013 /* ============================================
00014 I2Cdev device library code is placed under the MIT license
00015 Copyright (c) 2011 Jeff Rowberg
00016
00017 Permission is hereby granted, free of charge, to any person obtaining a copy
00018 of this software and associated documentation files (the "Software"), to deal
00019 in the Software without restriction, including without limitation the rights
00020 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
00021 copies of the Software, and to permit persons to whom the Software is
00022 furnished to do so, subject to the following conditions:
00023
00024 The above copyright notice and this permission notice shall be included in
00025 all copies or substantial portions of the Software.
00026
00027 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
00028 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
00029 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
00030 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
00031 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
00032 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
00033 THE SOFTWARE.
00034 ===============================================
00035 */
00036
00037 #ifndef _MPU6050_H_
00038 #define _MPU6050_H_
00039
00040 #include "I2Cdev.h"
00041 #include <avr/pgmspace.h>
00042
00043 #define MPU6050_ADDRESS_AD0_LOW     0x68 // address pin low (GND), default for InvenSense evaluation board
00044 #define MPU6050_ADDRESS_AD0_HIGH    0x69 // address pin high (VCC)
00045 #define MPU6050_DEFAULT_ADDRESS     MPU6050_ADDRESS_AD0_LOW
00046
00047 #define MPU6050_RA_XG_OFFS_TC       0x00 //[7] PWR_MODE, [6:1] XG_OFFS_TC, [0] OTP_BNK_VLD
00048 #define MPU6050_RA_YG_OFFS_TC       0x01 //[7] PWR_MODE, [6:1] YG_OFFS_TC, [0] OTP_BNK_VLD
00049 #define MPU6050_RA_ZG_OFFS_TC       0x02 //[7] PWR_MODE, [6:1] ZG_OFFS_TC, [0] OTP_BNK_VLD
00050 #define MPU6050_RA_X_FINE_GAIN      0x03 //[7:0] X_FINE_GAIN
00051 #define MPU6050_RA_Y_FINE_GAIN      0x04 //[7:0] Y_FINE_GAIN
00052 #define MPU6050_RA_Z_FINE_GAIN      0x05 //[7:0] Z_FINE_GAIN
00053 #define MPU6050_RA_XA_OFFS_H        0x06 //[15:0] XA_OFFS
00054 #define MPU6050_RA_XA_OFFS_L_TC     0x07
00055 #define MPU6050_RA_YA_OFFS_H        0x08 //[15:0] YA_OFFS
00056 #define MPU6050_RA_YA_OFFS_L_TC     0x09
00057 #define MPU6050_RA_ZA_OFFS_H        0x0A //[15:0] ZA_OFFS
00058 #define MPU6050_RA_ZA_OFFS_L_TC     0x0B
00059 #define MPU6050_RA_XG_OFFS_USRH     0x13 //[15:0] XG_OFFS_USR
00060 #define MPU6050_RA_XG_OFFS_USRL     0x14
00061 #define MPU6050_RA_YG_OFFS_USRH     0x15 //[15:0] YG_OFFS_USR
00062 #define MPU6050_RA_YG_OFFS_USRL     0x16
00063 #define MPU6050_RA_ZG_OFFS_USRH     0x17 //[15:0] ZG_OFFS_USR
00064 #define MPU6050_RA_ZG_OFFS_USRL     0x18
00065 #define MPU6050_RA_SMPLRT_DIV       0x19
00066 #define MPU6050_RA_CONFIG           0x1A
00067 #define MPU6050_RA_GYRO_CONFIG      0x1B
00068 #define MPU6050_RA_ACCEL_CONFIG     0x1C
00069 #define MPU6050_RA_FF_THR           0x1D
00070 #define MPU6050_RA_FF_DUR           0x1E
00071 #define MPU6050_RA_MOT_THR          0x1F
```

```
00072 #define MPU6050_RA_MOT_DUR          0x20
00073 #define MPU6050_RA_ZRMOT_THR        0x21
00074 #define MPU6050_RA_ZRMOT_DUR        0x22
00075 #define MPU6050_RA_FIFO_EN          0x23
00076 #define MPU6050_RA_I2C_MST_CTRL     0x24
00077 #define MPU6050_RA_I2C_SLV0_ADDR    0x25
00078 #define MPU6050_RA_I2C_SLV0_REG     0x26
00079 #define MPU6050_RA_I2C_SLV0_CTRL    0x27
00080 #define MPU6050_RA_I2C_SLV1_ADDR    0x28
00081 #define MPU6050_RA_I2C_SLV1_REG     0x29
00082 #define MPU6050_RA_I2C_SLV1_CTRL    0x2A
00083 #define MPU6050_RA_I2C_SLV2_ADDR    0x2B
00084 #define MPU6050_RA_I2C_SLV2_REG     0x2C
00085 #define MPU6050_RA_I2C_SLV2_CTRL    0x2D
00086 #define MPU6050_RA_I2C_SLV3_ADDR    0x2E
00087 #define MPU6050_RA_I2C_SLV3_REG     0x2F
00088 #define MPU6050_RA_I2C_SLV3_CTRL    0x30
00089 #define MPU6050_RA_I2C_SLV4_ADDR    0x31
00090 #define MPU6050_RA_I2C_SLV4_REG     0x32
00091 #define MPU6050_RA_I2C_SLV4_DO      0x33
00092 #define MPU6050_RA_I2C_SLV4_CTRL    0x34
00093 #define MPU6050_RA_I2C_SLV4_DI      0x35
00094 #define MPU6050_RA_I2C_MST_STATUS   0x36
00095 #define MPU6050_RA_INT_PIN_CFG      0x37
00096 #define MPU6050_RA_INT_ENABLE       0x38
00097 #define MPU6050_RA_DMP_INT_STATUS   0x39
00098 #define MPU6050_RA_INT_STATUS       0x3A
00099 #define MPU6050_RA_ACCEL_XOUT_H     0x3B
00100 #define MPU6050_RA_ACCEL_XOUT_L     0x3C
00101 #define MPU6050_RA_ACCEL_YOUT_H     0x3D
00102 #define MPU6050_RA_ACCEL_YOUT_L     0x3E
00103 #define MPU6050_RA_ACCEL_ZOUT_H     0x3F
00104 #define MPU6050_RA_ACCEL_ZOUT_L     0x40
00105 #define MPU6050_RA_TEMP_OUT_H       0x41
00106 #define MPU6050_RA_TEMP_OUT_L       0x42
00107 #define MPU6050_RA_GYRO_XOUT_H      0x43
00108 #define MPU6050_RA_GYRO_XOUT_L      0x44
00109 #define MPU6050_RA_GYRO_YOUT_H      0x45
00110 #define MPU6050_RA_GYRO_YOUT_L      0x46
00111 #define MPU6050_RA_GYRO_ZOUT_H      0x47
00112 #define MPU6050_RA_GYRO_ZOUT_L      0x48
00113 #define MPU6050_RA_EXT_SENS_DATA_00 0x49
00114 #define MPU6050_RA_EXT_SENS_DATA_01 0x4A
00115 #define MPU6050_RA_EXT_SENS_DATA_02 0x4B
00116 #define MPU6050_RA_EXT_SENS_DATA_03 0x4C
00117 #define MPU6050_RA_EXT_SENS_DATA_04 0x4D
00118 #define MPU6050_RA_EXT_SENS_DATA_05 0x4E
00119 #define MPU6050_RA_EXT_SENS_DATA_06 0x4F
00120 #define MPU6050_RA_EXT_SENS_DATA_07 0x50
00121 #define MPU6050_RA_EXT_SENS_DATA_08 0x51
00122 #define MPU6050_RA_EXT_SENS_DATA_09 0x52
00123 #define MPU6050_RA_EXT_SENS_DATA_10 0x53
00124 #define MPU6050_RA_EXT_SENS_DATA_11 0x54
00125 #define MPU6050_RA_EXT_SENS_DATA_12 0x55
00126 #define MPU6050_RA_EXT_SENS_DATA_13 0x56
00127 #define MPU6050_RA_EXT_SENS_DATA_14 0x57
00128 #define MPU6050_RA_EXT_SENS_DATA_15 0x58
00129 #define MPU6050_RA_EXT_SENS_DATA_16 0x59
00130 #define MPU6050_RA_EXT_SENS_DATA_17 0x5A
00131 #define MPU6050_RA_EXT_SENS_DATA_18 0x5B
00132 #define MPU6050_RA_EXT_SENS_DATA_19 0x5C
00133 #define MPU6050_RA_EXT_SENS_DATA_20 0x5D
00134 #define MPU6050_RA_EXT_SENS_DATA_21 0x5E
00135 #define MPU6050_RA_EXT_SENS_DATA_22 0x5F
00136 #define MPU6050_RA_EXT_SENS_DATA_23 0x60
00137 #define MPU6050_RA_MOT_DETECT_STATUS    0x61
00138 #define MPU6050_RA_I2C_SLV0_DO      0x63
00139 #define MPU6050_RA_I2C_SLV1_DO      0x64
00140 #define MPU6050_RA_I2C_SLV2_DO      0x65
00141 #define MPU6050_RA_I2C_SLV3_DO      0x66
00142 #define MPU6050_RA_I2C_MST_DELAY_CTRL   0x67
00143 #define MPU6050_RA_SIGNAL_PATH_RESET    0x68
00144 #define MPU6050_RA_MOT_DETECT_CTRL      0x69
00145 #define MPU6050_RA_USER_CTRL        0x6A
00146 #define MPU6050_RA_PWR_MGMT_1       0x6B
00147 #define MPU6050_RA_PWR_MGMT_2       0x6C
00148 #define MPU6050_RA_BANK_SEL         0x6D
00149 #define MPU6050_RA_MEM_START_ADDR   0x6E
00150 #define MPU6050_RA_MEM_R_W          0x6F
00151 #define MPU6050_RA_DMP_CFG_1        0x70
00152 #define MPU6050_RA_DMP_CFG_2        0x71
00153 #define MPU6050_RA_FIFO_COUNTH      0x72
00154 #define MPU6050_RA_FIFO_COUNTL      0x73
00155 #define MPU6050_RA_FIFO_R_W         0x74
00156 #define MPU6050_RA_WHO_AM_I         0x75
00157
00158 #define MPU6050_TC_PWR_MODE_BIT     7
00159 #define MPU6050_TC_OFFSET_BIT       6
```

```
00160 #define MPU6050_TC_OFFSET_LENGTH    6
00161 #define MPU6050_TC_OTP_BNK_VLD_BIT  0
00162
00163 #define MPU6050_VDDIO_LEVEL_VLOGIC  0
00164 #define MPU6050_VDDIO_LEVEL_VDD     1
00165
00166 #define MPU6050_CFG_EXT_SYNC_SET_BIT    5
00167 #define MPU6050_CFG_EXT_SYNC_SET_LENGTH 3
00168 #define MPU6050_CFG_DLPF_CFG_BIT    2
00169 #define MPU6050_CFG_DLPF_CFG_LENGTH 3
00170
00171 #define MPU6050_EXT_SYNC_DISABLED       0x0
00172 #define MPU6050_EXT_SYNC_TEMP_OUT_L     0x1
00173 #define MPU6050_EXT_SYNC_GYRO_XOUT_L    0x2
00174 #define MPU6050_EXT_SYNC_GYRO_YOUT_L    0x3
00175 #define MPU6050_EXT_SYNC_GYRO_ZOUT_L    0x4
00176 #define MPU6050_EXT_SYNC_ACCEL_XOUT_L   0x5
00177 #define MPU6050_EXT_SYNC_ACCEL_YOUT_L   0x6
00178 #define MPU6050_EXT_SYNC_ACCEL_ZOUT_L   0x7
00179
00180 #define MPU6050_DLPF_BW_256         0x00
00181 #define MPU6050_DLPF_BW_188         0x01
00182 #define MPU6050_DLPF_BW_98          0x02
00183 #define MPU6050_DLPF_BW_42          0x03
00184 #define MPU6050_DLPF_BW_20          0x04
00185 #define MPU6050_DLPF_BW_10          0x05
00186 #define MPU6050_DLPF_BW_5           0x06
00187
00188 #define MPU6050_GCONFIG_FS_SEL_BIT      4
00189 #define MPU6050_GCONFIG_FS_SEL_LENGTH   2
00190
00191 #define MPU6050_GYRO_FS_250         0x00
00192 #define MPU6050_GYRO_FS_500         0x01
00193 #define MPU6050_GYRO_FS_1000        0x02
00194 #define MPU6050_GYRO_FS_2000        0x03
00195
00196 #define MPU6050_ACONFIG_XA_ST_BIT           7
00197 #define MPU6050_ACONFIG_YA_ST_BIT           6
00198 #define MPU6050_ACONFIG_ZA_ST_BIT           5
00199 #define MPU6050_ACONFIG_AFS_SEL_BIT         4
00200 #define MPU6050_ACONFIG_AFS_SEL_LENGTH      2
00201 #define MPU6050_ACONFIG_ACCEL_HPF_BIT       2
00202 #define MPU6050_ACONFIG_ACCEL_HPF_LENGTH    3
00203
00204 #define MPU6050_ACCEL_FS_2          0x00
00205 #define MPU6050_ACCEL_FS_4          0x01
00206 #define MPU6050_ACCEL_FS_8          0x02
00207 #define MPU6050_ACCEL_FS_16         0x03
00208
00209 #define MPU6050_DHPF_RESET          0x00
00210 #define MPU6050_DHPF_5              0x01
00211 #define MPU6050_DHPF_2P5            0x02
00212 #define MPU6050_DHPF_1P25           0x03
00213 #define MPU6050_DHPF_0P63           0x04
00214 #define MPU6050_DHPF_HOLD           0x07
00215
00216 #define MPU6050_TEMP_FIFO_EN_BIT    7
00217 #define MPU6050_XG_FIFO_EN_BIT      6
00218 #define MPU6050_YG_FIFO_EN_BIT      5
00219 #define MPU6050_ZG_FIFO_EN_BIT      4
00220 #define MPU6050_ACCEL_FIFO_EN_BIT   3
00221 #define MPU6050_SLV2_FIFO_EN_BIT    2
00222 #define MPU6050_SLV1_FIFO_EN_BIT    1
00223 #define MPU6050_SLV0_FIFO_EN_BIT    0
00224
00225 #define MPU6050_MULT_MST_EN_BIT     7
00226 #define MPU6050_WAIT_FOR_ES_BIT     6
00227 #define MPU6050_SLV_3_FIFO_EN_BIT   5
00228 #define MPU6050_I2C_MST_P_NSR_BIT   4
00229 #define MPU6050_I2C_MST_CLK_BIT     3
00230 #define MPU6050_I2C_MST_CLK_LENGTH  4
00231
00232 #define MPU6050_CLOCK_DIV_348       0x0
00233 #define MPU6050_CLOCK_DIV_333       0x1
00234 #define MPU6050_CLOCK_DIV_320       0x2
00235 #define MPU6050_CLOCK_DIV_308       0x3
00236 #define MPU6050_CLOCK_DIV_296       0x4
00237 #define MPU6050_CLOCK_DIV_286       0x5
00238 #define MPU6050_CLOCK_DIV_276       0x6
00239 #define MPU6050_CLOCK_DIV_267       0x7
00240 #define MPU6050_CLOCK_DIV_258       0x8
00241 #define MPU6050_CLOCK_DIV_500       0x9
00242 #define MPU6050_CLOCK_DIV_471       0xA
00243 #define MPU6050_CLOCK_DIV_444       0xB
00244 #define MPU6050_CLOCK_DIV_421       0xC
00245 #define MPU6050_CLOCK_DIV_400       0xD
00246 #define MPU6050_CLOCK_DIV_381       0xE
00247 #define MPU6050_CLOCK_DIV_364       0xF
```

```
00248
00249 #define MPU6050_I2C_SLV_RW_BIT      7
00250 #define MPU6050_I2C_SLV_ADDR_BIT    6
00251 #define MPU6050_I2C_SLV_ADDR_LENGTH 7
00252 #define MPU6050_I2C_SLV_EN_BIT      7
00253 #define MPU6050_I2C_SLV_BYTE_SW_BIT 6
00254 #define MPU6050_I2C_SLV_REG_DIS_BIT 5
00255 #define MPU6050_I2C_SLV_GRP_BIT     4
00256 #define MPU6050_I2C_SLV_LEN_BIT     3
00257 #define MPU6050_I2C_SLV_LEN_LENGTH  4
00258
00259 #define MPU6050_I2C_SLV4_RW_BIT         7
00260 #define MPU6050_I2C_SLV4_ADDR_BIT       6
00261 #define MPU6050_I2C_SLV4_ADDR_LENGTH    7
00262 #define MPU6050_I2C_SLV4_EN_BIT         7
00263 #define MPU6050_I2C_SLV4_INT_EN_BIT     6
00264 #define MPU6050_I2C_SLV4_REG_DIS_BIT    5
00265 #define MPU6050_I2C_SLV4_MST_DLY_BIT    4
00266 #define MPU6050_I2C_SLV4_MST_DLY_LENGTH 5
00267
00268 #define MPU6050_MST_PASS_THROUGH_BIT    7
00269 #define MPU6050_MST_I2C_SLV4_DONE_BIT   6
00270 #define MPU6050_MST_I2C_LOST_ARB_BIT    5
00271 #define MPU6050_MST_I2C_SLV4_NACK_BIT   4
00272 #define MPU6050_MST_I2C_SLV3_NACK_BIT   3
00273 #define MPU6050_MST_I2C_SLV2_NACK_BIT   2
00274 #define MPU6050_MST_I2C_SLV1_NACK_BIT   1
00275 #define MPU6050_MST_I2C_SLV0_NACK_BIT   0
00276
00277 #define MPU6050_INTCFG_INT_LEVEL_BIT        7
00278 #define MPU6050_INTCFG_INT_OPEN_BIT         6
00279 #define MPU6050_INTCFG_LATCH_INT_EN_BIT     5
00280 #define MPU6050_INTCFG_INT_RD_CLEAR_BIT     4
00281 #define MPU6050_INTCFG_FSYNC_INT_LEVEL_BIT  3
00282 #define MPU6050_INTCFG_FSYNC_INT_EN_BIT     2
00283 #define MPU6050_INTCFG_I2C_BYPASS_EN_BIT    1
00284 #define MPU6050_INTCFG_CLKOUT_EN_BIT        0
00285
00286 #define MPU6050_INTMODE_ACTIVEHIGH  0x00
00287 #define MPU6050_INTMODE_ACTIVELOW   0x01
00288
00289 #define MPU6050_INTDRV_PUSHPULL     0x00
00290 #define MPU6050_INTDRV_OPENDRAIN    0x01
00291
00292 #define MPU6050_INTLATCH_50USPULSE  0x00
00293 #define MPU6050_INTLATCH_WAITCLEAR  0x01
00294
00295 #define MPU6050_INTCLEAR_STATUSREAD 0x00
00296 #define MPU6050_INTCLEAR_ANYREAD    0x01
00297
00298 #define MPU6050_INTERRUPT_FF_BIT            7
00299 #define MPU6050_INTERRUPT_MOT_BIT           6
00300 #define MPU6050_INTERRUPT_ZMOT_BIT          5
00301 #define MPU6050_INTERRUPT_FIFO_OFLOW_BIT    4
00302 #define MPU6050_INTERRUPT_I2C_MST_INT_BIT   3
00303 #define MPU6050_INTERRUPT_PLL_RDY_INT_BIT   2
00304 #define MPU6050_INTERRUPT_DMP_INT_BIT       1
00305 #define MPU6050_INTERRUPT_DATA_RDY_BIT      0
00306
00307 // TODO: figure out what these actually do
00308 // UMPL source code is not very obivous
00309 #define MPU6050_DMPINT_5_BIT            5
00310 #define MPU6050_DMPINT_4_BIT            4
00311 #define MPU6050_DMPINT_3_BIT            3
00312 #define MPU6050_DMPINT_2_BIT            2
00313 #define MPU6050_DMPINT_1_BIT            1
00314 #define MPU6050_DMPINT_0_BIT            0
00315
00316 #define MPU6050_MOTION_MOT_XNEG_BIT     7
00317 #define MPU6050_MOTION_MOT_XPOS_BIT     6
00318 #define MPU6050_MOTION_MOT_YNEG_BIT     5
00319 #define MPU6050_MOTION_MOT_YPOS_BIT     4
00320 #define MPU6050_MOTION_MOT_ZNEG_BIT     3
00321 #define MPU6050_MOTION_MOT_ZPOS_BIT     2
00322 #define MPU6050_MOTION_MOT_ZRMOT_BIT    0
00323
00324 #define MPU6050_DELAYCTRL_DELAY_ES_SHADOW_BIT   7
00325 #define MPU6050_DELAYCTRL_I2C_SLV4_DLY_EN_BIT   4
00326 #define MPU6050_DELAYCTRL_I2C_SLV3_DLY_EN_BIT   3
00327 #define MPU6050_DELAYCTRL_I2C_SLV2_DLY_EN_BIT   2
00328 #define MPU6050_DELAYCTRL_I2C_SLV1_DLY_EN_BIT   1
00329 #define MPU6050_DELAYCTRL_I2C_SLV0_DLY_EN_BIT   0
00330
00331 #define MPU6050_PATHRESET_GYRO_RESET_BIT    2
00332 #define MPU6050_PATHRESET_ACCEL_RESET_BIT   1
00333 #define MPU6050_PATHRESET_TEMP_RESET_BIT    0
00334
00335 #define MPU6050_DETECT_ACCEL_ON_DELAY_BIT       5
```

```
00336 #define MPU6050_DETECT_ACCEL_ON_DELAY_LENGTH    2
00337 #define MPU6050_DETECT_FF_COUNT_BIT             3
00338 #define MPU6050_DETECT_FF_COUNT_LENGTH          2
00339 #define MPU6050_DETECT_MOT_COUNT_BIT            1
00340 #define MPU6050_DETECT_MOT_COUNT_LENGTH         2
00341
00342 #define MPU6050_DETECT_DECREMENT_RESET  0x0
00343 #define MPU6050_DETECT_DECREMENT_1      0x1
00344 #define MPU6050_DETECT_DECREMENT_2      0x2
00345 #define MPU6050_DETECT_DECREMENT_4      0x3
00346
00347 #define MPU6050_USERCTRL_DMP_EN_BIT             7
00348 #define MPU6050_USERCTRL_FIFO_EN_BIT            6
00349 #define MPU6050_USERCTRL_I2C_MST_EN_BIT         5
00350 #define MPU6050_USERCTRL_I2C_IF_DIS_BIT         4
00351 #define MPU6050_USERCTRL_DMP_RESET_BIT          3
00352 #define MPU6050_USERCTRL_FIFO_RESET_BIT         2
00353 #define MPU6050_USERCTRL_I2C_MST_RESET_BIT      1
00354 #define MPU6050_USERCTRL_SIG_COND_RESET_BIT     0
00355
00356 #define MPU6050_PWR1_DEVICE_RESET_BIT   7
00357 #define MPU6050_PWR1_SLEEP_BIT          6
00358 #define MPU6050_PWR1_CYCLE_BIT          5
00359 #define MPU6050_PWR1_TEMP_DIS_BIT       3
00360 #define MPU6050_PWR1_CLKSEL_BIT         2
00361 #define MPU6050_PWR1_CLKSEL_LENGTH      3
00362
00363 #define MPU6050_CLOCK_INTERNAL          0x00
00364 #define MPU6050_CLOCK_PLL_XGYRO         0x01
00365 #define MPU6050_CLOCK_PLL_YGYRO         0x02
00366 #define MPU6050_CLOCK_PLL_ZGYRO         0x03
00367 #define MPU6050_CLOCK_PLL_EXT32K        0x04
00368 #define MPU6050_CLOCK_PLL_EXT19M        0x05
00369 #define MPU6050_CLOCK_KEEP_RESET        0x07
00370
00371 #define MPU6050_PWR2_LP_WAKE_CTRL_BIT       7
00372 #define MPU6050_PWR2_LP_WAKE_CTRL_LENGTH    2
00373 #define MPU6050_PWR2_STBY_XA_BIT            5
00374 #define MPU6050_PWR2_STBY_YA_BIT            4
00375 #define MPU6050_PWR2_STBY_ZA_BIT            3
00376 #define MPU6050_PWR2_STBY_XG_BIT            2
00377 #define MPU6050_PWR2_STBY_YG_BIT            1
00378 #define MPU6050_PWR2_STBY_ZG_BIT            0
00379
00380 #define MPU6050_WAKE_FREQ_1P25      0x0
00381 #define MPU6050_WAKE_FREQ_2P5       0x1
00382 #define MPU6050_WAKE_FREQ_5         0x2
00383 #define MPU6050_WAKE_FREQ_10        0x3
00384
00385 #define MPU6050_BANKSEL_PRFTCH_EN_BIT       6
00386 #define MPU6050_BANKSEL_CFG_USER_BANK_BIT   5
00387 #define MPU6050_BANKSEL_MEM_SEL_BIT         4
00388 #define MPU6050_BANKSEL_MEM_SEL_LENGTH      5
00389
00390 #define MPU6050_WHO_AM_I_BIT        6
00391 #define MPU6050_WHO_AM_I_LENGTH     6
00392
00393 #define MPU6050_DMP_MEMORY_BANKS        8
00394 #define MPU6050_DMP_MEMORY_BANK_SIZE    256
00395 #define MPU6050_DMP_MEMORY_CHUNK_SIZE   16
00396
00397 // note: DMP code memory blocks defined at end of header file
00398
00399 class MPU6050 {
00400     public:
00401         MPU6050();
00402         MPU6050(uint8_t address);
00403
00404         void initialize();
00405         bool testConnection();
00406
00407         // AUX_VDDIO register
00408         uint8_t getAuxVDDIOLevel();
00409         void setAuxVDDIOLevel(uint8_t level);
00410
00411         // SMPLRT_DIV register
00412         uint8_t getRate();
00413         void setRate(uint8_t rate);
00414
00415         // CONFIG register
00416         uint8_t getExternalFrameSync();
00417         void setExternalFrameSync(uint8_t sync);
00418         uint8_t getDLPFMode();
00419         void setDLPFMode(uint8_t bandwidth);
00420
00421         // GYRO_CONFIG register
00422         uint8_t getFullScaleGyroRange();
00423         void setFullScaleGyroRange(uint8_t range);
```

```
00424
00425         // ACCEL_CONFIG register
00426         bool getAccelXSelfTest();
00427         void setAccelXSelfTest(bool enabled);
00428         bool getAccelYSelfTest();
00429         void setAccelYSelfTest(bool enabled);
00430         bool getAccelZSelfTest();
00431         void setAccelZSelfTest(bool enabled);
00432         uint8_t getFullScaleAccelRange();
00433         void setFullScaleAccelRange(uint8_t range);
00434         uint8_t getDHPFMode();
00435         void setDHPFMode(uint8_t mode);
00436
00437         // FF_THR register
00438         uint8_t getFreefallDetectionThreshold();
00439         void setFreefallDetectionThreshold(uint8_t threshold);
00440
00441         // FF_DUR register
00442         uint8_t getFreefallDetectionDuration();
00443         void setFreefallDetectionDuration(uint8_t duration);
00444
00445         // MOT_THR register
00446         uint8_t getMotionDetectionThreshold();
00447         void setMotionDetectionThreshold(uint8_t threshold);
00448
00449         // MOT_DUR register
00450         uint8_t getMotionDetectionDuration();
00451         void setMotionDetectionDuration(uint8_t duration);
00452
00453         // ZRMOT_THR register
00454         uint8_t getZeroMotionDetectionThreshold();
00455         void setZeroMotionDetectionThreshold(uint8_t threshold);
00456
00457         // ZRMOT_DUR register
00458         uint8_t getZeroMotionDetectionDuration();
00459         void setZeroMotionDetectionDuration(uint8_t duration);
00460
00461         // FIFO_EN register
00462         bool getTempFIFOEnabled();
00463         void setTempFIFOEnabled(bool enabled);
00464         bool getXGyroFIFOEnabled();
00465         void setXGyroFIFOEnabled(bool enabled);
00466         bool getYGyroFIFOEnabled();
00467         void setYGyroFIFOEnabled(bool enabled);
00468         bool getZGyroFIFOEnabled();
00469         void setZGyroFIFOEnabled(bool enabled);
00470         bool getAccelFIFOEnabled();
00471         void setAccelFIFOEnabled(bool enabled);
00472         bool getSlave2FIFOEnabled();
00473         void setSlave2FIFOEnabled(bool enabled);
00474         bool getSlave1FIFOEnabled();
00475         void setSlave1FIFOEnabled(bool enabled);
00476         bool getSlave0FIFOEnabled();
00477         void setSlave0FIFOEnabled(bool enabled);
00478
00479         // I2C_MST_CTRL register
00480         bool getMultiMasterEnabled();
00481         void setMultiMasterEnabled(bool enabled);
00482         bool getWaitForExternalSensorEnabled();
00483         void setWaitForExternalSensorEnabled(bool enabled);
00484         bool getSlave3FIFOEnabled();
00485         void setSlave3FIFOEnabled(bool enabled);
00486         bool getSlaveReadWriteTransitionEnabled();
00487         void setSlaveReadWriteTransitionEnabled(bool enabled);
00488         uint8_t getMasterClockSpeed();
00489         void setMasterClockSpeed(uint8_t speed);
00490
00491         // I2C_SLV* registers (Slave 0-3)
00492         uint8_t getSlaveAddress(uint8_t num);
00493         void setSlaveAddress(uint8_t num, uint8_t address);
00494         uint8_t getSlaveRegister(uint8_t num);
00495         void setSlaveRegister(uint8_t num, uint8_t reg);
00496         bool getSlaveEnabled(uint8_t num);
00497         void setSlaveEnabled(uint8_t num, bool enabled);
00498         bool getSlaveWordByteSwap(uint8_t num);
00499         void setSlaveWordByteSwap(uint8_t num, bool enabled);
00500         bool getSlaveWriteMode(uint8_t num);
00501         void setSlaveWriteMode(uint8_t num, bool mode);
00502         bool getSlaveWordGroupOffset(uint8_t num);
00503         void setSlaveWordGroupOffset(uint8_t num, bool enabled);
00504         uint8_t getSlaveDataLength(uint8_t num);
00505         void setSlaveDataLength(uint8_t num, uint8_t length);
00506
00507         // I2C_SLV* registers (Slave 4)
00508         uint8_t getSlave4Address();
00509         void setSlave4Address(uint8_t address);
00510         uint8_t getSlave4Register();
00511         void setSlave4Register(uint8_t reg);
```

```
00512         void setSlave4OutputByte(uint8_t data);
00513         bool getSlave4Enabled();
00514         void setSlave4Enabled(bool enabled);
00515         bool getSlave4InterruptEnabled();
00516         void setSlave4InterruptEnabled(bool enabled);
00517         bool getSlave4WriteMode();
00518         void setSlave4WriteMode(bool mode);
00519         uint8_t getSlave4MasterDelay();
00520         void setSlave4MasterDelay(uint8_t delay);
00521         uint8_t getSlate4InputByte();
00522
00523         // I2C_MST_STATUS register
00524         bool getPassthroughStatus();
00525         bool getSlave4IsDone();
00526         bool getLostArbitration();
00527         bool getSlave4Nack();
00528         bool getSlave3Nack();
00529         bool getSlave2Nack();
00530         bool getSlave1Nack();
00531         bool getSlave0Nack();
00532
00533         // INT_PIN_CFG register
00534         bool getInterruptMode();
00535         void setInterruptMode(bool mode);
00536         bool getInterruptDrive();
00537         void setInterruptDrive(bool drive);
00538         bool getInterruptLatch();
00539         void setInterruptLatch(bool latch);
00540         bool getInterruptLatchClear();
00541         void setInterruptLatchClear(bool clear);
00542         bool getFSyncInterruptLevel();
00543         void setFSyncInterruptLevel(bool level);
00544         bool getFSyncInterruptEnabled();
00545         void setFSyncInterruptEnabled(bool enabled);
00546         bool getI2CBypassEnabled();
00547         void setI2CBypassEnabled(bool enabled);
00548         bool getClockOutputEnabled();
00549         void setClockOutputEnabled(bool enabled);
00550
00551         // INT_ENABLE register
00552         bool getIntFreefallEnabled();
00553         void setIntFreefallEnabled(bool enabled);
00554         bool getIntMotionEnabled();
00555         void setIntMotionEnabled(bool enabled);
00556         bool getIntZeroMotionEnabled();
00557         void setIntZeroMotionEnabled(bool enabled);
00558         bool getIntFIFOBufferOverflowEnabled();
00559         void setIntFIFOBufferOverflowEnabled(bool enabled);
00560         bool getIntI2CMasterEnabled();
00561         void setIntI2CMasterEnabled(bool enabled);
00562         bool getIntDataReadyEnabled();
00563         void setIntDataReadyEnabled(bool enabled);
00564
00565         // INT_STATUS register
00566         bool getIntFreefallStatus();
00567         bool getIntMotionStatus();
00568         bool getIntZeroMotionStatus();
00569         bool getIntFIFOBufferOverflowStatus();
00570         bool getIntI2CMasterStatus();
00571         bool getIntDataReadyStatus();
00572
00573         // ACCEL_*OUT_* registers
00574         void getMotion9(int16_t* ax, int16_t* ay, int16_t* az, int16_t* gx, int16_t* gy, int16_t* gz, int16_t* mx, int16_t* my, int16
_t* mz);
00575         void getMotion6(int16_t* ax, int16_t* ay, int16_t* az, int16_t* gx, int16_t* gy, int16_t* gz);
00576         void getAcceleration(int16_t* x, int16_t* y, int16_t* z);
00577         int16_t getAccelerationX();
00578         int16_t getAccelerationY();
00579         int16_t getAccelerationZ();
00580
00581         // TEMP_OUT_* registers
00582         int16_t getTemperature();
00583
00584         // GYRO_*OUT_* registers
00585         void getRotation(int16_t* x, int16_t* y, int16_t* z);
00586         int16_t getRotationX();
00587         int16_t getRotationY();
00588         int16_t getRotationZ();
00589
00590         // EXT_SENS_DATA_* registers
00591         uint8_t getExternalSensorByte(int position);
00592         uint16_t getExternalSensorWord(int position);
00593         uint32_t getExternalSensorDWord(int position);
00594
00595         // MOT_DETECT_STATUS register
00596         bool getXNegMotionDetected();
00597         bool getXPosMotionDetected();
00598         bool getYNegMotionDetected();
```

```
00599         bool getYPosMotionDetected();
00600         bool getZNegMotionDetected();
00601         bool getZPosMotionDetected();
00602         bool getZeroMotionDetected();
00603
00604         // I2C_SLV*_DO register
00605         void setSlaveOutputByte(uint8_t num, uint8_t data);
00606
00607         // I2C_MST_DELAY_CTRL register
00608         bool getExternalShadowDelayEnabled();
00609         void setExternalShadowDelayEnabled(bool enabled);
00610         bool getSlaveDelayEnabled(uint8_t num);
00611         void setSlaveDelayEnabled(uint8_t num, bool enabled);
00612
00613         // SIGNAL_PATH_RESET register
00614         void resetGyroscopePath();
00615         void resetAccelerometerPath();
00616         void resetTemperaturePath();
00617
00618         // MOT_DETECT_CTRL register
00619         uint8_t getAccelerometerPowerOnDelay();
00620         void setAccelerometerPowerOnDelay(uint8_t delay);
00621         uint8_t getFreefallDetectionCounterDecrement();
00622         void setFreefallDetectionCounterDecrement(uint8_t decrement);
00623         uint8_t getMotionDetectionCounterDecrement();
00624         void setMotionDetectionCounterDecrement(uint8_t decrement);
00625
00626         // USER_CTRL register
00627         bool getFIFOEnabled();
00628         void setFIFOEnabled(bool enabled);
00629         bool getI2CMasterModeEnabled();
00630         void setI2CMasterModeEnabled(bool enabled);
00631         void switchSPIEnabled(bool enabled);
00632         void resetFIFO();
00633         void resetI2CMaster();
00634         void resetSensors();
00635
00636         // PWR_MGMT_1 register
00637         void reset();
00638         bool getSleepEnabled();
00639         void setSleepEnabled(bool enabled);
00640         bool getWakeCycleEnabled();
00641         void setWakeCycleEnabled(bool enabled);
00642         bool getTempSensorEnabled();
00643         void setTempSensorEnabled(bool enabled);
00644         uint8_t getClockSource();
00645         void setClockSource(uint8_t source);
00646
00647         // PWR_MGMT_2 register
00648         uint8_t getWakeFrequency();
00649         void setWakeFrequency(uint8_t frequency);
00650         bool getStandbyXAccelEnabled();
00651         void setStandbyXAccelEnabled(bool enabled);
00652         bool getStandbyYAccelEnabled();
00653         void setStandbyYAccelEnabled(bool enabled);
00654         bool getStandbyZAccelEnabled();
00655         void setStandbyZAccelEnabled(bool enabled);
00656         bool getStandbyXGyroEnabled();
00657         void setStandbyXGyroEnabled(bool enabled);
00658         bool getStandbyYGyroEnabled();
00659         void setStandbyYGyroEnabled(bool enabled);
00660         bool getStandbyZGyroEnabled();
00661         void setStandbyZGyroEnabled(bool enabled);
00662
00663         // FIFO_COUNT_* registers
00664         uint16_t getFIFOCount();
00665
00666         // FIFO_R_W register
00667         uint8_t getFIFOByte();
00668         void setFIFOByte(uint8_t data);
00669
00670         // WHO_AM_I register
00671         uint8_t getDeviceID();
00672         void setDeviceID(uint8_t id);
00673
00674         // ======== UNDOCUMENTED/DMP REGISTERS/METHODS ========
00675
00676         // XG_OFFS_TC register
00677         int8_t getXGyroOffset();
00678         void setXGyroOffset(int8_t offset);
00679
00680         // YG_OFFS_TC register
00681         int8_t getYGyroOffset();
00682         void setYGyroOffset(int8_t offset);
00683
00684         // ZG_OFFS_TC register
00685         int8_t getZGyroOffset();
00686         void setZGyroOffset(int8_t offset);
```

```
00687
00688         // X_FINE_GAIN register
00689         int8_t getXFineGain();
00690         void setXFineGain(int8_t gain);
00691
00692         // Y_FINE_GAIN register
00693         int8_t getYFineGain();
00694         void setYFineGain(int8_t gain);
00695
00696         // Z_FINE_GAIN register
00697         int8_t getZFineGain();
00698         void setZFineGain(int8_t gain);
00699
00700         // XA_OFFS_* registers
00701         int16_t getXAccelOffset();
00702         void setXAccelOffset(int16_t offset);
00703
00704         // YA_OFFS_* register
00705         int16_t getYAccelOffset();
00706         void setYAccelOffset(int16_t offset);
00707
00708         // ZA_OFFS_* register
00709         int16_t getZAccelOffset();
00710         void setZAccelOffset(int16_t offset);
00711
00712         // XG_OFFS_USR* registers
00713         int16_t getXGyroOffsetUser();
00714         void setXGyroOffsetUser(int16_t offset);
00715
00716         // YG_OFFS_USR* register
00717         int16_t getYGyroOffsetUser();
00718         void setYGyroOffsetUser(int16_t offset);
00719
00720         // ZG_OFFS_USR* register
00721         int16_t getZGyroOffsetUser();
00722         void setZGyroOffsetUser(int16_t offset);
00723
00724         // INT_ENABLE register (DMP functions)
00725         bool getIntPLLReadyEnabled();
00726         void setIntPLLReadyEnabled(bool enabled);
00727         bool getIntDMPEnabled();
00728         void setIntDMPEnabled(bool enabled);
00729
00730         // DMP_INT_STATUS
00731         bool getDMPInt5Status();
00732         bool getDMPInt4Status();
00733         bool getDMPInt3Status();
00734         bool getDMPInt2Status();
00735         bool getDMPInt1Status();
00736         bool getDMPInt0Status();
00737
00738         // INT_STATUS register (DMP functions)
00739         bool getIntPLLReadyStatus();
00740         bool getIntDMPStatus();
00741
00742         // USER_CTRL register (DMP functions)
00743         bool getDMPEnabled();
00744         void setDMPEnabled(bool enabled);
00745         void resetDMP();
00746
00747         // BANK_SEL register
00748         void setMemoryBank(uint8_t bank, bool prefetchEnabled=false, bool userBank=false);
00749
00750         // MEM_START_ADDR register
00751         void setMemoryStartAddress(uint8_t address);
00752
00753         // MEM_R_W register
00754         uint8_t readMemoryByte();
00755         void writeMemoryByte(uint8_t data);
00756         void readMemoryBlock(uint8_t *data, uint16_t dataSize, uint8_t bank=0, uint8_t address=0);
00757         bool writeMemoryBlock(uint8_t *data, uint16_t dataSize, uint8_t bank=0, uint8_t address=0, bool verify=true, bool useProgMem=
false);
00758         bool writeProgMemoryBlock(uint8_t *data, uint16_t dataSize, uint8_t bank=0, uint8_t address=0, bool verify=true);
00759
00760         // DMP_CFG_1 register
00761         uint8_t getDMPConfig1();
00762         void setDMPConfig1(uint8_t config);
00763
00764         // DMP_CFG_2 register
00765         uint8_t getDMPConfig2();
00766         void setDMPConfig2(uint8_t config);
00767
00768     private:
00769         uint8_t devAddr;
00770         uint8_t buffer[14];
00771 };
00772
00773 #ifdef MPU6050_INCLUDE_DMP_MOTIONAPPS20
```

```
00774     /* This is only included if you want it, since it eats about 2K of program
00775      * memory, which is a waste if you aren't using the DMP (or if you aren't
00776      * using this particular flavor of DMP).
00777      *
00778      * Source is from the InvenSense MotionApps v2 demo code. Original source is
00779      * unavailable, unless you happen to be amazing as decompiling binary by
00780      * hand (in which case, please contact me, and I'm totally serious).
00781      *
00782      * Also, I'd like to offer many, many thanks to Noah Zerkin for all of the
00783      * DMP reverse-engineering he did to help make this bit of wizardry
00784      * possible.
00785      */
00786
00787     #define MPU6050_DMP_CODE_SIZE 1929
00788
00789     // this block of memory gets written to the MPU on start-up, and it seems
00790     // to be volatile memory, so it has to be done each time (it only takes ~1
00791     // second though)
00792     prog_uchar dmpMemory[MPU6050_DMP_CODE_SIZE] PROGMEM = {
00793         // bank 0, 256 bytes
00794         0xFB, 0x00, 0x00, 0x3E, 0x00, 0x0B, 0x00, 0x36, 0x00, 0x01, 0x00, 0x02, 0x00, 0x03, 0x00, 0x00,
00795         0x00, 0x65, 0x00, 0x54, 0xFF, 0xEF, 0x00, 0x00, 0xFA, 0x80, 0x00, 0x0B, 0x12, 0x82, 0x00, 0x01,
00796         0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00797         0x00, 0x28, 0x00, 0x00, 0xFF, 0xFF, 0x45, 0x81, 0xFF, 0xFF, 0xFA, 0x72, 0x00, 0x00, 0x00, 0x00,
00798         0x00, 0x00, 0x03, 0xE8, 0x00, 0x00, 0x00, 0x01, 0x00, 0x01, 0x7F, 0xFF, 0xFF, 0xFE, 0x80, 0x01,
00799         0x00, 0x1B, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00800         0x00, 0x3E, 0x03, 0x30, 0x40, 0x00, 0x00, 0x00, 0x02, 0xCA, 0xE3, 0x09, 0x3E, 0x80, 0x00, 0x00,
00801         0x20, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00, 0x60, 0x00, 0x00, 0x00,
00802         0x41, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x0B, 0x2A, 0x00, 0x00, 0x16, 0x55, 0x00, 0x00, 0x21, 0x82,
00803         0xFD, 0x87, 0x26, 0x50, 0xFD, 0x80, 0x00, 0x00, 0x00, 0x1F, 0x00, 0x00, 0x00, 0x05, 0x80, 0x00,
00804         0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00,
00805         0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x04, 0x6F, 0x00, 0x02, 0x65, 0x32, 0x00, 0x00, 0x5E, 0xC0,
00806         0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00807         0xFB, 0x8C, 0x6F, 0x5D, 0xFD, 0x5D, 0x08, 0xD9, 0x00, 0x7C, 0x73, 0x3B, 0x00, 0x6C, 0x12, 0xCC,
00808         0x32, 0x00, 0x13, 0x9D, 0x32, 0x00, 0xD0, 0xD6, 0x32, 0x00, 0x08, 0x00, 0x40, 0x00, 0x01, 0xF4,
00809         0xFF, 0xE6, 0x80, 0x79, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0xD0, 0xD6, 0x00, 0x00, 0x27, 0x10,
00810
00811         // bank 1, 256 bytes
00812         0xFB, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00813         0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x01, 0x00, 0x00, 0x00,
00814         0x00, 0x00, 0xFA, 0x36, 0xFF, 0xBC, 0x30, 0x8E, 0x00, 0x05, 0xFB, 0xF0, 0xFF, 0xD9, 0x5B, 0xC8,
00815         0xFF, 0xD0, 0x9A, 0xBE, 0x00, 0x00, 0x10, 0xA9, 0xFF, 0xF4, 0x1E, 0xB2, 0x00, 0xCE, 0xBB, 0xF7,
00816         0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x02, 0x00, 0x02, 0x02, 0x00, 0x00, 0x0C,
00817         0xFF, 0xC2, 0x80, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0xCF, 0x80, 0x00, 0x40, 0x00, 0x00, 0x00,
00818         0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00, 0x00, 0x00, 0x14,
00819         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00820         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00821         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00822         0x00, 0x00, 0x00, 0x00, 0x03, 0x3F, 0x68, 0xB6, 0x79, 0x35, 0x28, 0xBC, 0xC6, 0x7E, 0xD1, 0x6C,
00823         0x80, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0xB2, 0x6A, 0x00, 0x00, 0x00, 0x00,
00824         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3F, 0xF0, 0x00, 0x00, 0x00, 0x30,
00825         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00826         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00827         0x00, 0x00, 0x25, 0x4D, 0x00, 0x2F, 0x70, 0x6D, 0x00, 0x00, 0x05, 0xAE, 0x00, 0x0C, 0x02, 0xD0,
00828
00829         // bank 2, 256 bytes
00830         0x00, 0x00, 0x00, 0x00, 0x00, 0x65, 0x00, 0x54, 0xFF, 0xEF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00831         0x00, 0x00, 0x01, 0x00, 0x00, 0x44, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00, 0x00, 0x00, 0x01, 0x00,
00832         0x00, 0x00, 0x00, 0x00, 0x00, 0x65, 0x00, 0x00, 0x00, 0x54, 0x00, 0x00, 0xFF, 0xEF, 0x00, 0x00,
00833         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00834         0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00835         0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00836         0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00837         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00838         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00839         0x00, 0x1B, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00840         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00841         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00,
00842         0x00, 0x1B, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00843         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00844         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00845         0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
00846
00847         // bank 3, 256 bytes
00848         0xD8, 0xDC, 0xBA, 0xA2, 0xF1, 0xDE, 0xB2, 0xB8, 0xB4, 0xA8, 0x81, 0x91, 0xF7, 0x4A, 0x90, 0x7F,
00849         0x91, 0x6A, 0xF3, 0xF9, 0xDB, 0xA8, 0xF9, 0xB0, 0xBA, 0xA0, 0x80, 0xF2, 0xCE, 0x81, 0xF3, 0xC2,
00850         0xF1, 0xC1, 0xF2, 0xC3, 0xF3, 0xCC, 0xA2, 0xB2, 0x80, 0xF1, 0xC6, 0xD8, 0x80, 0xBA, 0xA7, 0xDF,
00851         0xDF, 0xDF, 0xF2, 0xA7, 0xC3, 0xCB, 0xC5, 0xB6, 0xF0, 0x87, 0xA2, 0x94, 0x24, 0x48, 0x70, 0x3C,
00852         0x95, 0x40, 0x68, 0x34, 0x58, 0x9B, 0x78, 0xA2, 0xF1, 0x83, 0x92, 0x2D, 0x55, 0x7D, 0xD8, 0xB1,
00853         0xB4, 0xB8, 0xA1, 0xD0, 0x91, 0x80, 0xF2, 0x70, 0xF3, 0x70, 0xF2, 0x7C, 0x80, 0xA8, 0xF1, 0x01,
00854         0xB0, 0x98, 0x87, 0xD9, 0x43, 0xD8, 0x86, 0xC9, 0x88, 0xBA, 0xA1, 0xF2, 0x0E, 0xB8, 0x97, 0x80,
00855         0xF1, 0xA9, 0xDF, 0xDF, 0xDF, 0xAA, 0xDF, 0xDF, 0xDF, 0xF2, 0xAA, 0xC5, 0xCD, 0xC7, 0xA9, 0x0C,
00856         0xC9, 0x2C, 0x97, 0x97, 0x97, 0x97, 0xF1, 0xA9, 0x89, 0x26, 0x46, 0x66, 0xB0, 0xB4, 0xBA, 0x80,
00857         0xAC, 0xDE, 0xF2, 0xCA, 0xF1, 0xB2, 0x8C, 0x02, 0xA9, 0xB6, 0x98, 0x00, 0x89, 0x0E, 0x16, 0x1E,
00858         0xB8, 0xA9, 0xB4, 0x99, 0x2C, 0x54, 0x7C, 0xB0, 0x8A, 0xA8, 0x96, 0x36, 0x56, 0x76, 0xF1, 0xB9,
00859         0xAF, 0xB4, 0xB0, 0x83, 0xC0, 0xB8, 0xA8, 0x97, 0x11, 0xB1, 0x8F, 0x98, 0xB9, 0xAF, 0xF0, 0x24,
00860         0x08, 0x44, 0x10, 0x64, 0x18, 0xF1, 0xA3, 0x29, 0x55, 0x7D, 0xAF, 0x83, 0xB5, 0x93, 0xAF, 0xF0,
00861         0x00, 0x28, 0x50, 0xF1, 0xA3, 0x86, 0x9F, 0x61, 0xA6, 0xDA, 0xDE, 0xDF, 0xD9, 0xFA, 0xA3, 0x86,
```

```
00862         0x96, 0xDB, 0x31, 0xA6, 0xD9, 0xF8, 0xDF, 0xBA, 0xA6, 0x8F, 0xC2, 0xC5, 0xC7, 0xB2, 0x8C, 0xC1,
00863         0xB8, 0xA2, 0xDF, 0xDF, 0xDF, 0xA3, 0xDF, 0xDF, 0xDF, 0xD8, 0xD8, 0xF1, 0xB8, 0xA8, 0xB2, 0x86,
00864
00865         // bank 4, 256 bytes
00866         0xB4, 0x98, 0x0D, 0x35, 0x5D, 0xB8, 0xAA, 0x98, 0xB0, 0x87, 0x2D, 0x35, 0x3D, 0xB2, 0xB6, 0xBA,
00867         0xAF, 0x8C, 0x96, 0x19, 0x8F, 0x9F, 0xA7, 0x0E, 0x16, 0x1E, 0xB4, 0x9A, 0xB8, 0xAA, 0x87, 0x2C,
00868         0x54, 0x7C, 0xB9, 0xA3, 0xDE, 0xDF, 0xDF, 0xA3, 0xB1, 0x80, 0xF2, 0xC4, 0xCD, 0xC9, 0xF1, 0xB8,
00869         0xA9, 0xB4, 0x99, 0x83, 0x0D, 0x35, 0x5D, 0x89, 0xB9, 0xA3, 0x2D, 0x55, 0x7D, 0xB5, 0x93, 0xA3,
00870         0x0E, 0x16, 0x1E, 0xA9, 0x2C, 0x54, 0x7C, 0xB8, 0xB4, 0xB0, 0xF1, 0x97, 0x83, 0xA8, 0x11, 0x84,
00871         0xA5, 0x09, 0x98, 0xA3, 0x83, 0xF0, 0xDA, 0x24, 0x08, 0x44, 0x10, 0x64, 0x18, 0xD8, 0xF1, 0xA5,
00872         0x29, 0x55, 0x7D, 0xA5, 0x85, 0x95, 0x02, 0x1A, 0x2E, 0x3A, 0x56, 0x5A, 0x40, 0x48, 0xF9, 0xF3,
00873         0xA3, 0xD9, 0xF8, 0xF0, 0x98, 0x83, 0x24, 0x08, 0x44, 0x10, 0x64, 0x18, 0x97, 0x82, 0xA8, 0xF1,
00874         0x11, 0xF0, 0x98, 0xA2, 0x24, 0x08, 0x44, 0x10, 0x64, 0x18, 0xDA, 0xF3, 0xDE, 0xD8, 0x83, 0xA5,
00875         0x94, 0x01, 0xD9, 0xA3, 0x02, 0xF1, 0xA2, 0xC3, 0xC5, 0xC7, 0xD8, 0xF1, 0x84, 0x92, 0xA2, 0x4D,
00876         0xDA, 0x2A, 0xD8, 0x48, 0x69, 0xD9, 0x2A, 0xD8, 0x68, 0x55, 0xDA, 0x32, 0xD8, 0x50, 0x71, 0xD9,
00877         0x32, 0xD8, 0x70, 0x5D, 0xDA, 0x3A, 0xD8, 0x58, 0x79, 0xD9, 0x3A, 0xD8, 0x78, 0x93, 0xA3, 0x4D,
00878         0xDA, 0x2A, 0xD8, 0x48, 0x69, 0xD9, 0x2A, 0xD8, 0x68, 0x55, 0xDA, 0x32, 0xD8, 0x50, 0x71, 0xD9,
00879         0x32, 0xD8, 0x70, 0x5D, 0xDA, 0x3A, 0xD8, 0x58, 0x79, 0xD9, 0x3A, 0xD8, 0x78, 0xA8, 0x8A, 0x9A,
00880         0xF0, 0x28, 0x50, 0x78, 0x9E, 0xF3, 0x88, 0x18, 0xF1, 0x9F, 0x1D, 0x98, 0xA8, 0xD9, 0x08, 0xD8,
00881         0xC8, 0x9F, 0x12, 0x9E, 0xF3, 0x15, 0xA8, 0xDA, 0x12, 0x10, 0xD8, 0xF1, 0xAF, 0xC8, 0x97, 0x87,
00882
00883         // bank 5, 256 bytes
00884         0x34, 0xB5, 0xB9, 0x94, 0xA4, 0x21, 0xF3, 0xD9, 0x22, 0xD8, 0xF2, 0x2D, 0xF3, 0xD9, 0x2A, 0xD8,
00885         0xF2, 0x35, 0xF3, 0xD9, 0x32, 0xD8, 0x81, 0xA4, 0x60, 0x60, 0x61, 0xD9, 0x61, 0xD8, 0x6C, 0x68,
00886         0x69, 0xD9, 0x69, 0xD8, 0x74, 0x70, 0x71, 0xD9, 0x71, 0xD8, 0xB1, 0xA3, 0x84, 0x19, 0x3D, 0x5D,
00887         0xA3, 0x83, 0x1A, 0x3E, 0x5E, 0x93, 0x10, 0x30, 0x81, 0x10, 0x11, 0xB8, 0xB0, 0xAF, 0x8F, 0x94,
00888         0xF2, 0xDA, 0x3E, 0xD8, 0xB4, 0x9A, 0xA8, 0x87, 0x29, 0xDA, 0xF8, 0xD8, 0x87, 0x9A, 0x35, 0xDA,
00889         0xF8, 0xD8, 0x87, 0x9A, 0x3D, 0xDA, 0xF8, 0xD8, 0xB1, 0xB9, 0xA4, 0x98, 0x85, 0x02, 0x2E, 0x56,
00890         0xA5, 0x81, 0x00, 0x0C, 0x14, 0xA3, 0x97, 0xB0, 0x8A, 0xF1, 0x2D, 0xD9, 0x28, 0xD8, 0x4D, 0xD9,
00891         0x48, 0xD8, 0x6D, 0xD9, 0x68, 0xD8, 0xB1, 0x84, 0x0D, 0xDA, 0x0E, 0xD8, 0xA3, 0x29, 0x83, 0xDA,
00892         0x2C, 0x0E, 0xD8, 0xA3, 0x84, 0x49, 0x83, 0xDA, 0x2C, 0x4C, 0x0E, 0xD8, 0xB8, 0xB0, 0xA8, 0x8A,
00893         0x9A, 0xF5, 0x20, 0xAA, 0xDA, 0xDF, 0xD8, 0xA8, 0x40, 0xAA, 0xD0, 0xDA, 0xDE, 0xD8, 0xA8, 0x60,
00894         0xAA, 0xDA, 0xD0, 0xDF, 0xD8, 0xF1, 0x97, 0x86, 0xA8, 0x31, 0x9B, 0x06, 0x99, 0x07, 0xAB, 0x97,
00895         0x28, 0x88, 0x9B, 0xF0, 0x0C, 0x20, 0x14, 0x40, 0xB8, 0xB0, 0xB4, 0xA8, 0x8C, 0x9C, 0xF0, 0x04,
00896         0x28, 0x51, 0x79, 0x1D, 0x30, 0x14, 0x38, 0xB2, 0x82, 0xAB, 0xD0, 0x98, 0x2C, 0x50, 0x50, 0x78,
00897         0x78, 0x9B, 0xF1, 0x1A, 0xB0, 0xF0, 0x8A, 0x9C, 0xA8, 0x29, 0x51, 0x79, 0x8B, 0x29, 0x51, 0x79,
00898         0x8A, 0x24, 0x70, 0x59, 0x8B, 0x20, 0x58, 0x71, 0x8A, 0x44, 0x69, 0x38, 0x8B, 0x39, 0x40, 0x68,
00899         0x8A, 0x64, 0x48, 0x31, 0x8B, 0x30, 0x49, 0x60, 0xA5, 0x88, 0x20, 0x09, 0x71, 0x58, 0x44, 0x68,
00900
00901         // bank 6, 256 bytes
00902         0x11, 0x39, 0x64, 0x49, 0x30, 0x19, 0xF1, 0xAC, 0x00, 0x2C, 0x54, 0x7C, 0xF0, 0x8C, 0xA8, 0x04,
00903         0x28, 0x50, 0x78, 0xF1, 0x88, 0x97, 0x26, 0xA8, 0x59, 0x98, 0xAC, 0x8C, 0x02, 0x26, 0x46, 0x66,
00904         0xF0, 0x89, 0x9C, 0xA8, 0x29, 0x51, 0x79, 0x24, 0x70, 0x59, 0x44, 0x69, 0x38, 0x64, 0x48, 0x31,
00905         0xA9, 0x88, 0x09, 0x20, 0x59, 0x70, 0xAB, 0x11, 0x38, 0x40, 0x69, 0xA8, 0x19, 0x31, 0x48, 0x60,
00906         0x8C, 0xA8, 0x3C, 0x41, 0x5C, 0x20, 0x7C, 0x00, 0xF1, 0x87, 0x98, 0x19, 0x86, 0xA8, 0x6E, 0x76,
00907         0x7E, 0xA9, 0x99, 0x88, 0x2D, 0x55, 0x7D, 0x9E, 0xB9, 0xA3, 0x8A, 0x22, 0x8A, 0x6E, 0x8A, 0x56,
00908         0x8A, 0x5E, 0x9F, 0xB1, 0x83, 0x06, 0x26, 0x46, 0x66, 0x0E, 0x2E, 0x4E, 0x6E, 0x9D, 0xB8, 0xAD,
00909         0x00, 0x2C, 0x54, 0x7C, 0xF2, 0xB1, 0x8C, 0xB4, 0x99, 0xB9, 0xA3, 0x2D, 0x55, 0x7D, 0x81, 0x91,
00910         0xAC, 0x38, 0xAD, 0x3A, 0xB5, 0x83, 0x91, 0xAC, 0x2D, 0xD9, 0x28, 0xD8, 0x4D, 0xD9, 0x48, 0xD8,
00911         0x6D, 0xD9, 0x68, 0xD8, 0x8C, 0x9D, 0xAE, 0x29, 0xD9, 0x04, 0xAE, 0xD8, 0x51, 0xD9, 0x04, 0xAE,
00912         0xD8, 0x79, 0xD9, 0x04, 0xD8, 0x81, 0xF3, 0x9D, 0xAD, 0x00, 0x8D, 0xAE, 0x19, 0x81, 0xAD, 0xD9,
00913         0x01, 0xD8, 0xF2, 0xAE, 0xDA, 0x26, 0xD8, 0x8E, 0x91, 0x29, 0x83, 0xA7, 0xD9, 0xAD, 0xAD, 0xAD,
00914         0xAD, 0xF3, 0x2A, 0xD8, 0xD8, 0xF1, 0xB0, 0xAC, 0x89, 0x91, 0x3E, 0x5E, 0x76, 0xF3, 0xAC, 0x2E,
00915         0x2E, 0xF1, 0xB1, 0x8C, 0x5A, 0x9C, 0xAC, 0x2C, 0x28, 0x28, 0x28, 0x9C, 0xAC, 0x30, 0x18, 0xA8,
00916         0x98, 0x81, 0x28, 0x34, 0x3C, 0x97, 0x24, 0xA7, 0x28, 0x34, 0x3C, 0x9C, 0x24, 0xF2, 0xB0, 0x89,
00917         0xAC, 0x91, 0x2C, 0x4C, 0x6C, 0x8A, 0x9B, 0x2D, 0xD9, 0xD8, 0xD8, 0x51, 0xD9, 0xD8, 0xD8, 0x79,
00918
00919         // bank 7, 138 bytes (remainder)
00920         0xD9, 0xD8, 0xD8, 0xF1, 0x9E, 0x88, 0xA3, 0x31, 0xDA, 0xD8, 0xD8, 0x91, 0x2D, 0xD9, 0x28, 0xD8,
00921         0x4D, 0xD9, 0x48, 0xD8, 0x6D, 0xD9, 0x68, 0xD8, 0xB1, 0x83, 0x93, 0x35, 0x3D, 0x80, 0x25, 0xDA,
00922         0xD8, 0xD8, 0x85, 0x69, 0xDA, 0xD8, 0xD8, 0xB4, 0x93, 0x81, 0xA3, 0x28, 0x34, 0x3C, 0xF3, 0xAB,
00923         0x8B, 0xF8, 0xA3, 0x91, 0xB6, 0x09, 0xB4, 0xD9, 0xAB, 0xDE, 0xFA, 0xB0, 0x87, 0x9C, 0xB9, 0xA3,
00924         0xDD, 0xF1, 0xA3, 0xA3, 0xA3, 0xA3, 0x95, 0xF1, 0xA3, 0xA3, 0xA3, 0x9D, 0xF1, 0xA3, 0xA3, 0xA3,
00925         0xA3, 0xF2, 0xA3, 0xB4, 0x90, 0x80, 0xF2, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3,
00926         0xA3, 0xB2, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3, 0xA3, 0xB0, 0x87, 0xB5, 0x99, 0xF1, 0xA3, 0xA3, 0xA3,
00927         0x98, 0xF1, 0xA3, 0xA3, 0xA3, 0x97, 0xA3, 0xA3, 0xA3, 0xA3, 0xF3, 0x9B, 0xA3, 0xA3, 0xDC,
00928         0xB9, 0xA7, 0xF1, 0x26, 0x26, 0x26, 0xD8, 0xD8, 0xFF
00929     };
00930
00931     uint8_t dmpUpdates[29][9] = {
00932         { 0x03, 0x7B, 0x03, 0x4C, 0xCD, 0x6C },         // FCFG_1 inv_set_gyro_calibration
00933         { 0x03, 0xAB, 0x03, 0x36, 0x56, 0x76 },         // FCFG_3 inv_set_gyro_calibration
00934         { 0x00, 0x68, 0x04, 0x02, 0xCB, 0x47, 0xA2 },   // D_0_104 inv_set_gyro_calibration
00935         { 0x02, 0x18, 0x04, 0x00, 0x05, 0x8B, 0xC1 },   // D_0_24 inv_set_gyro_calibration
00936         { 0x01, 0x0C, 0x04, 0x00, 0x00, 0x00, 0x00 },   // D_1_152 inv_set_accel_calibration
00937         { 0x03, 0x7F, 0x06, 0x0C, 0xC9, 0x2C, 0x97, 0x97, 0x97 }, // FCFG_2 inv_set_accel_calibration
00938         { 0x03, 0x89, 0x03, 0x26, 0x46, 0x66 },         // FCFG_7 inv_set_accel_calibration
00939         { 0x00, 0x6C, 0x02, 0x20, 0x00 },               // D_0_108 inv_set_accel_calibration
00940         { 0x02, 0x40, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_00 inv_set_compass_calibration
00941         { 0x02, 0x44, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_01
00942         { 0x02, 0x48, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_02
00943         { 0x02, 0x4C, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_10
00944         { 0x02, 0x50, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_11
00945         { 0x02, 0x54, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_12
00946         { 0x02, 0x58, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_20
00947         { 0x02, 0x5C, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_21
00948         { 0x02, 0xBC, 0x04, 0x00, 0x00, 0x00, 0x00 },   // CPASS_MTX_22
00949         { 0x01, 0xEC, 0x04, 0x00, 0x00, 0x40, 0x00 },   // D_1_236 inv_apply_endian_accel
```

```
00950         { 0x03, 0x7F, 0x06, 0x0C, 0xC9, 0x2C, 0x97, 0x97, 0x97 }, // FCFG_2 inv_set_mpu_sensors
00951         { 0x04, 0x02, 0x03, 0x0D, 0x35, 0x5D },         // CFG_MOTION_BIAS inv_turn_on_bias_from_no_motion
00952         { 0x04, 0x09, 0x04, 0x87, 0x2D, 0x35, 0x3D },   // FCFG_5 inv_set_bias_update
00953         { 0x00, 0xA3, 0x01, 0x00 },                     // D_0_163 inv_set_dead_zone
00954         // SET INT_ENABLE at i=22
00955         { 0x07, 0x86, 0x01, 0xFE },                     // CFG_6 inv_set_fifo_interupt
00956         { 0x07, 0x41, 0x05, 0xF1, 0x20, 0x28, 0x30, 0x38 }, // CFG_8 inv_send_quaternion
00957         { 0x07, 0x7E, 0x01, 0x30 },                     // CFG_16 inv_set_footer
00958         { 0x07, 0x46, 0x01, 0x9A },                     // CFG_GYRO_SOURCE inv_send_gyro
00959         { 0x07, 0x47, 0x04, 0xF1, 0x28, 0x30, 0x38 },   // CFG_9 inv_send_gyro -> inv_construct3_fifo
00960         { 0x07, 0x6C, 0x04, 0xF1, 0x28, 0x30, 0x38 },   // CFG_12 inv_send_accel -> inv_construct3_fifo
00961         { 0x02, 0x16, 0x02, 0x00, 0x0A }                // D_0_22 inv_set_fifo_rate
00962 };
00963
00964 #endif
00965
00966 #endif /* _MPU6050_H_ */
```