simpleTouch™

API

Version 1.2

Revisions

Date	Change/Update	Version
6/6/2022	Initial Release	1.0
6/7/2022	Updated Example	1.1
6/11/2022	Corrected memory allocation from int to char	1.2

Table of Content

For who is this document?

This document describes the API for the simpleTouch™ software module and its implementation. It is for developers who have a working knowledge of PCAP touch screens, PCAP touch screen algorithms in touch ICs, and are implementing simpleTouch.

Overview

simpleTouch is a software module for PCAP touch ICs. It ingests untuned raw analog data from the touch screen, processes the data to determine a touch condition, and calculates the centroid for one or more (up to ten) touch points. By eliminating the need to tune PCAP touch screens for different substrates thicknesses, simpleTouch removes one of the most complex and time-consuming steps of implementing PCAP touchscreens.

simpleTouch sells as a software module pre-programmed at the touch IC maker's 3rd-party IC packaging house. Any developer (the touch IC maker or its customers) using a touch IC with simpleTouch can access the simpleTouch module through the simpleTouch API.

Core Blocks

simpleTouch consists of two components. An ANSI C header file and a binary blob containing the precompiled simpleTouch function. The binary blob is pre-programmed into the flash of the touch IC.

The header file contains the data structures and the prototype functions required to interact with simpleTouch.

Using simpleTouch

The simpleTouch module comes pre-programmed on touch ICs enabled with simpleTouch. To use simpleTouch, configure the simpleTouch module once at startup, then call the simpleTouch function after every scan of the touch screen. The simpleTouch function accepts raw data from the touch screen scan, processes the raw data, and returns a touch status with up to ten touch points. See data flow in Figure 1.

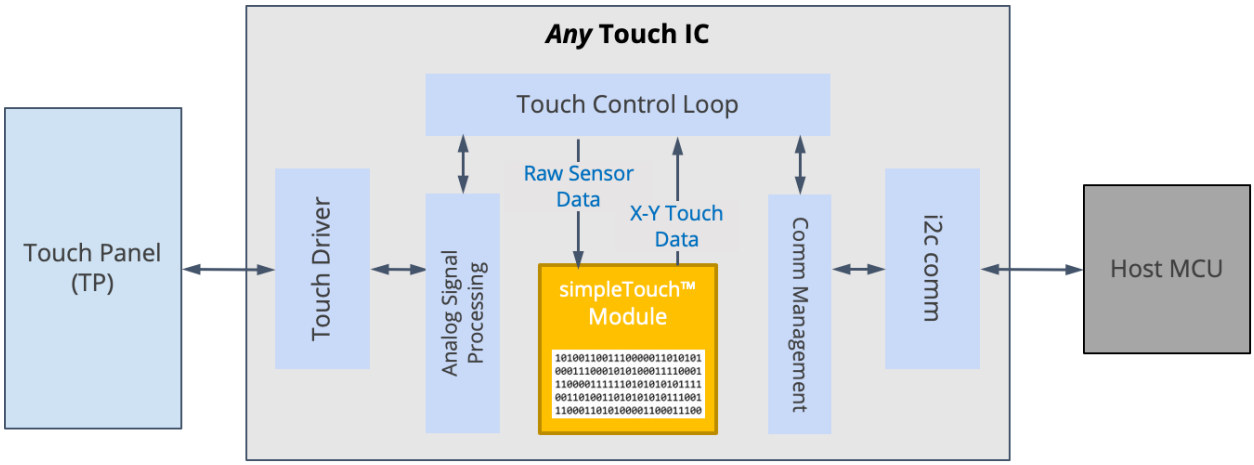

Figure 1 - Data Flow diagram for simpleTouch

Implementing simpleTouch

simpleTouch easily integrates into typical touch screen algorithms, see Figure 2 below.

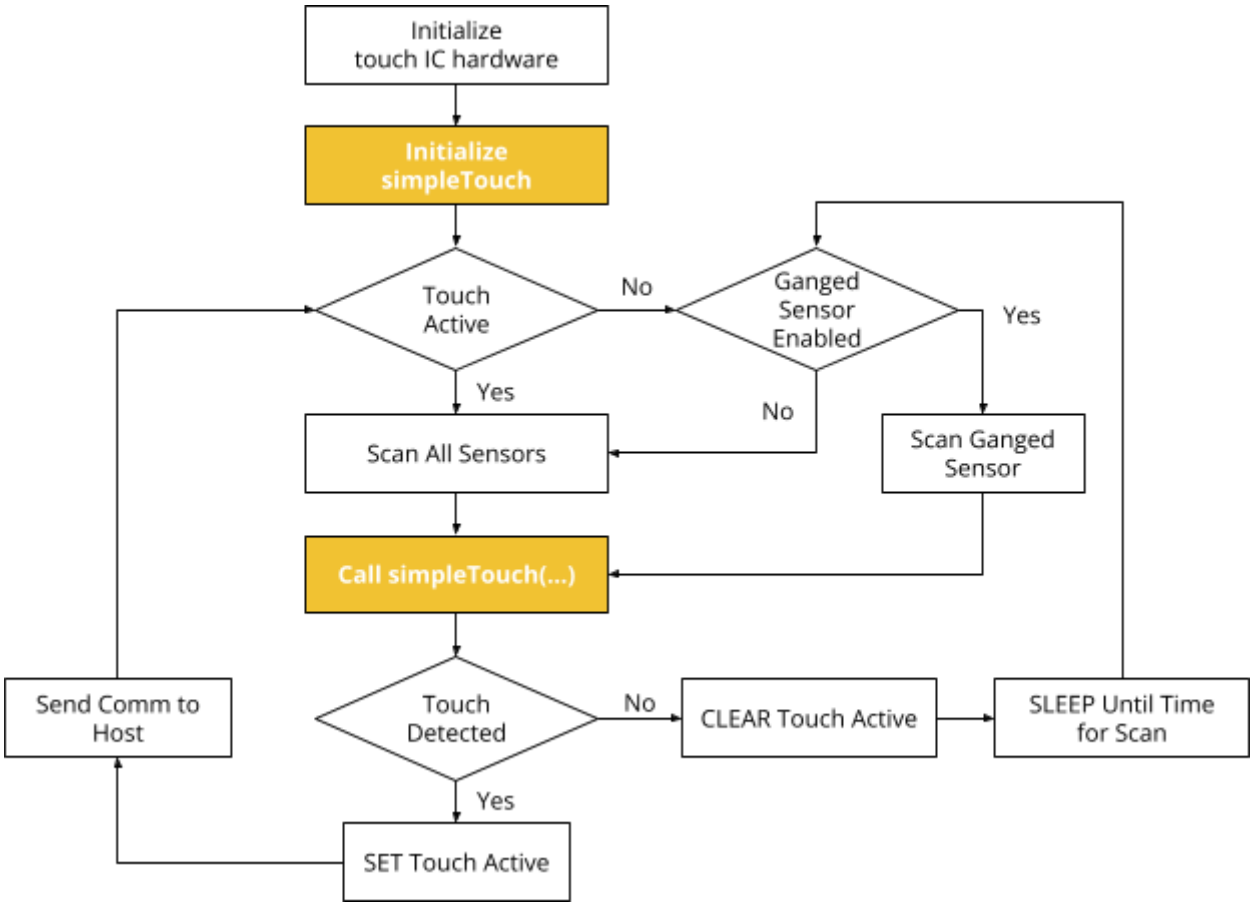

Figure 2 - A typical implementation of simpleTouch

simpleTouch.h

```
/*******************************************************************************
* File Name: simpleTouch.h
*
* Version: 0.3
*
* Description:
*  This file contains simpleTouch related function prototypes and constants.
*
********************************************************************************
* Copyright 2020-2022, Venntis, LLC.  All rights reserved.
* You may use this file only in accordance with the license, terms, conditions,
* disclaimers, and limitations in the end user license agreement accompanying
* the software package with which this file was provided.
*******************************************************************************/

#ifndef SIMPLETOUCHV2_SIMPLETOUCH_H
#define SIMPLETOUCHV2_SIMPLETOUCH_H

/***************************************
*       Definitions
***************************************/
enum scan_type{e_Normal, e_Slow, e_Fast};
enum touchscreen_type{e_Self, e_Mutual};


/*ControlRegister bit definitions */
#define TOUCHACTIVE 0x01      // SET = Touch detected, CLEAR = No touch detected
#define GANGEDSCAN 0x02       // SET = Ganged scanning, CLEAR = Normal scanning
#define GANGSCANCAPABLE 0x03  // SET = Capable of gang scanning, CLEAR = Not capable of gang
scanning

/***************************************
*       Structures to be declared by
*       user for us by simpleTouch
***************************************/
typedef struct
{
   unsigned char   uc_ScreenType;      // e_Self or e_Mutual
   unsigned char   uc_NumberOfRows;    // Number of Rows
   unsigned char   uc_NumberOfColumns; // Number of Columns
   unsigned int    xPixelRange;        // Pixel Range of display x dimension
   unsigned int    yPixelRange;        // Pixel Range of display y dimension
   unsigned int    ControlRegister;    // Control register Control bits and Status bits

   unsigned char*   Reserved;          // Reserved

} struct_simpleTouch_Configuation;

typedef struct
{
   unsigned int *uiA_RawSensorData;    // Array of Raw Sensor Data. Self:
GangSensor->Rows->Cols. Mutual:Top-Left -> Bottom-Right
                                    // [0] = gang sensor (leave unused if no gang sensor),
[1->n] = rows, [n+1->m] = columns
```

```
} struct_RawSensorData;

typedef struct
{
   signed char sc_status;              // e_NoTouch or e_Touch
   unsigned char uc_NumOfTouches;      // Number of touchs detected
   unsigned int uiA_touchpoint[10][2]; // Array of upto 10 touch points [touchPoint 0-9][x 0],
[touchPoint 0-9][y 1]
} struct_TouchResult;

/*************************************
*       Function Prototypes
*************************************/
void simpletouch(struct_simpleTouch_Configuation *simpleTouch_Config,   /* Pointer to
structure that holds the configuration values and working RAM for simpleTouch */
                 struct_RawSensorData *RawSensorData,                   /* Pointer to structure
that holds raw values from the latest scan */
                 struct_TouchResult *TouchResults);                    /* Pointer to structure
to save touch results */


#endif //SIMPLETOUCHV2_SIMPLETOUCH_H
```

Example

```
/*******************************************************************************
* File Name: main.c
*
* Version: 0.3
*
* Author: Bahar Wadia
*
* Description:
*  This file contains simpleTouch related function prototypes and constants.
*
********************************************************************************
* Copyright 2020-2022, Venntis, LLC.  All rights reserved.
* You may use this file only in accordance with the license, terms, conditions,
* disclaimers, and limitations in the end user license agreement accompanying
* the software package with which this file was provided.
*******************************************************************************/

#include <unistd.h>
#include "simpleTouch.h"


//Function prototypes
void f_usSleepTimer(unsigned long);
int8_t initializeTouchIC();
int8_t scanGangedSensor(struct_RawSensorData *RawSensorData);
int8_t scanAllSensors(struct_RawSensorData *RawSensorData);
int8_t sendComm(struct_TouchResult *TouchResults);

int main()
{

   // Declare simpleTouch structures
   struct_simpleTouch_Configuation simpleTouch_Config;
   struct_RawSensorData RawSensorData;
   struct_TouchResult TouchResults;

   // Set up the touch IC
   initializeTouchIC();

   //Configure simpleTouch
   simpleTouch_Config.uc_ScreenType = e_Self;              // self CAP touchscreen
   simpleTouch_Config.uc_NumberOfRows = 7;                 // Seven rows
   simpleTouch_Config.uc_NumberOfColumns = 8;              // Eight columns
   simpleTouch_Config.xPixelRange = 390;                   // 390 pixels in x dimension
   simpleTouch_Config.yPixelRange = 456;                   // 456 pixels in y dimension
   simpleTouch_Config.ControlRegister &= !TOUCHACTIVE;     // CLEAR touch status
   simpleTouch_Config.ControlRegister |= GANGSCANCAPABLE;  // SET capable of gang scanning
   simpleTouch_Config.ControlRegister &= !GANGEDSCAN;      // CLEAR ganged scan status

   unsigned char Reserved[(unsigned char) ((23 +
((1+simpleTouch_Config.uc_NumberOfRows+simpleTouch_Config.uc_NumberOfColumns)*20)) * 1.1)];
   simpleTouch_Config.Reserved = Reserved;
```

```
   // Allocate RAM for the raw data that simpleTouch will process
   unsigned int uiA_RawSensorData[1 + simpleTouch_Config.uc_NumberOfRows
   + simpleTouch_Config.uc_NumberOfColumns]; // [0] = gang sensor, [1->7] = rows, [8->15] =
columns

   // Assign the pointer to the raw data to the structure.
   RawSensorData.uiA_RawSensorData = uiA_RawSensorData;

   // Sensor scan loop
   while(1)
   {
       // Check if touch screen is touched
       if (simpleTouch_Config.ControlRegister & TOUCHACTIVE)
       {
           //Touch screen touched
           simpleTouch_Config.ControlRegister &= !GANGEDSCAN;     // CLEAR ganged scan status
           scanAllSensors(&RawSensorData); // Scan all sensors and return the RAW data in
RawSensorData


           simpletouch(&simpleTouch_Config, &RawSensorData, &TouchResults);
           if (simpleTouch_Config.ControlRegister & TOUCHACTIVE)
           {
               //Touch screen touched
               sendComm(&TouchResults);
           }
       }
       else
       {
           //Touch screen not touched
           if (simpleTouch_Config.ControlRegister & GANGSCANCAPABLE)
           {
               //if touch screen scanned as a ganged sensor
               simpleTouch_Config.ControlRegister |= GANGEDSCAN;     // SET ganged scan status
               scanGangedSensor(&RawSensorData); // Scan the ganged sensors and return the RAW
data in RawSensorData

           }
           else
           {
               simpleTouch_Config.ControlRegister &= !GANGEDSCAN;     // CLEAR ganged scan
status
               scanAllSensors(&RawSensorData); // Scan all sensors and return the RAW data in
RawSensorData

           }

           simpletouch(&simpleTouch_Config, &RawSensorData, &TouchResults);
           if (!(simpleTouch_Config.ControlRegister & TOUCHACTIVE))
           {
               f_usSleepTimer(1000);
           }
       }
   }

   return 1;
}
```

```
void f_usSleepTimer(unsigned long uSeconds)
{
   // Your code below this line
   usleep(uSeconds);
}

int8_t initializeTouchIC()
{
   // Your code below this line
   return 1;
}

int8_t scanGangedSensor(struct_RawSensorData *RawSensorData)
{
   // Your code below this line
   return 1;
}

int8_t scanAllSensors(struct_RawSensorData *RawSensorData)
{
   // Your code below this line
   return 1;
}

int8_t sendComm(struct_TouchResult *TouchResults)
{
   // Your code below this line
   return 1;
}
```