|  |
| --- |
| **深圳普得技术有限公司** |
| Pilot Robot IFTTT Data Structure Defination  Specification |
| Revision 0.1  Nov. 1st, 2015 |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | By | Description |
| V0.1  V0.2 | 11/01/15 | JY  JY | Temp: Preliminary Draft |
| V0.3 |  | JY |  |
| V0.4 |  | JY |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

© 2015 by Pilot Labs  
All rights reserved   
Printed in the U.S.A.

Pilot Labs is a registered trademarks of PILOT LABS, Inc.

THE DRAWINGS AND INFORMATION CONTAINED IN THIS DATA SHEET ARE THE EXCLUSIVE PROPERTY OF PILOT LABS INC. THE INFORMATION, DRAWINGS AND DESIGN CONCEPTS CONTAINED HEREIN ARE CONFIDENTIAL / PROPRIETARY. THEY SHALL BE MAINTAINED IN STRICT CONFIDENCE AND SHALL NOT BE RELEASED TO ANY THIRD PARTY WITHOUT THE EXPRESS WRITTEN PERMISSION OF PILOT LABS INC.

Table of Contents

1 简介 **错误！未定义书签。**

2 ID **错误！未定义书签。**

2.1 基本要求 错误！未定义书签。

2.2 机械动作 错误！未定义书签。

2.3 电源接口和其他表面接口 错误！未定义书签。

List Of Figures

**未找到图形项目表。**

List of Tables

未找到图形项目表。

# Introduction

Due to complexity of the Robot project, the IFTTT architecture is defined in order to quickly update the functions of the Robot. The firmware contains a list of input structures, output structures, system status structures, and IFTTT table.

The input structures contains all interrupt (INT) and generated input information from system status update, i.e. update system time and generate an INT to see of the system should respond.

The output structures contains all executions for the firmware, such as "reply voice: xxx", "drive motor to xxx", etc.

The system status structures contains current system information.

The IFTTT table contains all configurations of what input should cause what output.

The goals is for engineering team to gather all input and output, and let marketing to update and maintain IFTTT list (**not necessarily in C code**), so that later on the response can be updated easily.

The following is a proposed architecture, illustrates how system should work.



# Data Structures

## INPUT struct

"Input Structure" contains the following fields:

**name**: (description of the input)

**class**: (system, WiFi, Network, Zigbee Co, Zigbee Device, SD card, Voice, PressKey???, Power, time, GIS, cloud command, local sensor, etc.)

**subclass**:

**static\_or\_dynamic**: (INT trigger or static state change)

**priority**: (1-8, 1 being the highest priority)

**TBD1**:

**TBD2**:

The following is an example of input structure:

input(4356)=struct( ...

'name', {{'INT Voice Matched: "good moorning"'}}, ...

'class',{{'INT'}}, ...

'subclass',{{'INT Voice Matched'}}, ...

'static\_or\_dynamic',{{'dynamic'}}, ...

'priority',{{'1'}}, ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

Input structures is 13bit address, with [0:4095]. The following table defines input class and subclass:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | class | subclass | description | Special note |
| 1 | System (1024)  [0:511] | Firmware(64)  [0:63] | All firmware version or firmware sub block change generated input | i.e. firmware version needs update. |
| Hardware(64)  [64:127] | All hardware component changes generated input | i.e. one of hardware didn't function |
| Usage(16)  [128:143] | usage conditions caused input | i.e. "reset-to-factory" cause input to trigger beep |
| Work\_mode(16)  [256:271] | Work mode change caused input | i.e. work mode change cause notification to other blocks |
| Power(16)  [320:335] | Power mode change caused input | i.e. from "battery" to "power connected" cause notification to other blocks |
|  |  |  |
| Selftest(128)  [336:463] |  |  |
| Error(64)  [464:511] | System Error cause input notification |  |
| 2 | WiFi(128)  [512:639] | Config(64)  [512:575] | Config change cause input to other blocks |  |
| Physical(64)  [576:639] | Physical change cause input to other blocks |  |
|  |  |  |  |  |
| 3 | Network（128）  [640:767] | Config(64)  [640:703] | Config change cause input to other blocks |  |
| Status(64)  [704:767] | Status change cause input to other blocks |  |
|  |  |  |  |  |
| 4 | Zigbee Co(16)  [768:783] | Firmware(8) |  |  |
| Physical (8) |  |  |
|  |  |  |  |  |
| 5 | Zigbee Device(192)  [784:975] | Sensor status (64) | Sensor Status change, including new device join, device drop, RF signal power weak, etc. | All sensors |
| Sensor Alert (64) | Sensor alert: INT\_Sensor | Switch, light, lock, curtain, etc. |
| Command(64) | Zigbee command receive from Zigbee devices. |  |
|  |  |  |  |  |
| 6 | SD card(48)  [976:1023] | Volume (4)  [976:979] | Volume change. including unplug SD card. |  |
| Content(28)  [980:1023] | Content change in SD card |  |
|  |  |  |  |  |
| 7 | Time(16)  [1024:1039] | Local(8)  [1024:1031] | Input to change system time  **Scheduled time reached** | Update in 1 minute |
| Worldwide(8)  [1032:1039] | Input to change World time, time zone, etc.  **Scheduled time reached** | Update in 1 minute |
|  |  |  |  |  |
| 8 | Local sensor and input(24)  [1040:1063] | PressKey(8)  [1040:1047] | Input from press key (play/pause/volume up/volume down/forward/backward) |  |
| Gsensor(4)  [1048:1051] | INT of Gsensor and Gsensor reading |  |
| Touch Sensor(4)  [1052:1055] | INT of Touch Sensor (TBD) |  |
| PIR(4)  [1056:1059] | INT of Local PIR |  |
| Light Sensor(4)  [1060:1063] | INT of Light Sensor and Light intensity reading |  |
|  |  |  |  |  |
| 9 | GIS(16)  [1064:1079] | Worldwide(8)  [1064:1071] | Physical location detected worldwide changed cause input |  |
| In house(8)  [1072:1079] | Physical location in house changed cause input |  |
|  |  |  |  |  |
| 10 | Local Command  [1080:1175] | received(92)  [1080:1175] | Local command received |  |
|  |  |  |
|  |  |  |
|  |  |  |  |  |
| 11 | Cloud Command(64)  [1176:1239] | Received(92)  [1176:1239] | Remote command received |  |
|  |  |  |
|  |  |  |
|  |  |  |  |  |
| 12 | Voice(1832)  [1240:4095] | Local Command(256)  [1240:1495] | Received voice match in local command set |  |
| Expression(256)  [1496:1751] | A expression sound match, i.e. laugh, cry, etc. |  |
| Sound(256)  [1752:2007] | A sound match, i.e. water drop, hammer, etc. |  |
| Cloud matched command(2088)  [2008:4095] | A voice match in pre-define word or sentence. |  |
|  |  |  |  |  |

## status structure

"status structure" contains the following fields:

**name**: (description of status)

**class**: (system, WiFi, Network, Zigbee Co, Zigbee Device, SD card, Voice, Time, GIS, Cloud Command, IFTTT, Local Sensor and input, User Information)

**subclass**:

**value**:

**TBD1**:

**TBD2**:

The following is an example of status structure:

status(101)=struct( ...

'name', {{'system date'}}, ...

'class',{{'time'}}, ...

'subclass',{{'local'}}, ...

'value',[date], ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

status offset is 12bit address, [0:2047]. The following table defines status class and subclass:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | class | subclass | description | Special note |
| 1 | System (1024)  [0:511] | Firmware(64)  [0:63] | All firmware component details and status: "ready to use or not" |  |
| Hardware(64)  [64:127] | All hardware component details, and status: "ready to use or not" |  |
| Usage(16)  [128:143] | Define use conditions such as "reset-to-factory", etc. |  |
| Work\_mode(16)  [256:271] | Define "standby", "work", etc. work mode |  |
| Power(16)  [320:335] | Define "power connected" "charging" "battery %" etc. |  |
|  |  |  |
| Selftest(128)  [336:463] | Self Test Pass and Fail |  |
| Error(64)  [464:511] | System Error Code |  |
| 2 | WiFi(128)  [512:639] | Config(64)  [512:575] | WiFi Configuration | May support multiple WiFi config. |
| Physical(64)  [576:639] | WiFi physical layer information, such as signal strength, etc. |  |
|  |  |  |  |  |
| 3 | Network（128）  [640:767] | Config(64)  [640:703] | Network configuration information |  |
| Status(64)  [704:767] | Indicator network connection to all servers: WeChat, iFly, Pilot Labs, 3rd party story pool, etc. |  |
|  |  |  |  |  |
| 4 | Zigbee Co(16)  [768:783] | Firmware(8) | Zigbee Coordinator firmware information |  |
| Physical (8) | Define such as TX power , etc. |  |
|  |  |  |  |  |
| 5 | Zigbee Device(192)  [784:975] | Sensor(64) | Sensor map, including type, MAC address, signal strength, etc. | All sensors |
| Cntl(64) | Control device map, including type, MAC address, signal strength, etc. | Switch, light, lock, curtain, etc. |
| Comm(64) | Communication device map, including type, MAC address, signal strength, etc. | Remote controller, etc. |
|  |  |  |  |  |
| 6 | SD card(48)  [976:1023] | Volume (4)  [976:979] | Define SD card volume, usage. "0" means SD card not present or not accessible |  |
| Content(28)  [980:1023] | Update Content list on SD card (TBD) |  |
|  |  |  |  |  |
| 7 | Time(16)  [1024:1039] | Local(8)  [1024:1031] | Local time, date, weekdate, etc. | Update in 1 minute |
| Worldwide(8)  [1032:1039] | World time, time zone, etc. | Update in 1 minute |
|  |  |  |  |  |
| 8 | Local sensor and input(24)  [1040:1063] | PressKey(8)  [1040:1047] | The last key pressed (play/pause/volume up/volume down/forward/backward) |  |
| Gsensor(4)  [1048:1051] | INT of Gsensor and Gsensor reading |  |
| Touch Sensor(4)  [1052:1055] | INT of Touch Sensor (TBD) |  |
| PIR(4)  [1056:1059] | INT of Local PIR |  |
| Light Sensor(4)  [1060:1063] | INT of Light Sensor and Light intensity reading |  |
|  |  |  |  |  |
| 9 | GIS(16)  [1064:1079] | Worldwide(8)  [1064:1071] | Physical location detected worldwide (i.e. via IP address) |  |
| In house(8)  [1072:1079] | Physical location detected in house | Location distinguish in house and outside |
|  |  |  |  |  |
| 10 | Local Command  [1080:1175] | Monitor(64)  [1080:1143] | Local program execution monitor |  |
| Received(16)  [1144:1159] | Received system command. At least 8. |  |
| Executed(16)  [1160:1175] | Executed system command. At least 8. |  |
|  |  |  |  |  |
| 11 | Cloud Command(64)  [1176:1239] | Monitor(32)  [1176:1207] | All cloud function monitor, i.e. "cannot execute cloud commands", "not getting response", etc. |  |
| Received(16)  [1208:1223] | Store successfully received cloud commands. At least 8 | Some IFTTT command may base on previous Cloud command |
| Sent(16)  [1224:1239] | Store successfully sent cloud commands. At least 8 | Some IFTTT command may base on previous Cloud command |
|  |  |  |  |  |
| 12 | Voice(64)  [1240:1303] | Monitor(32)  [1240:1271] | All voice function monitor: recognition, reading, answering, etc. |  |
| Received(16)  [1272:1287] | Store successfully received voice command. At least 8 commands | **Some IFTTT will base on previous voice commands** |
| Sent(16)  [1288:1303] | Store voice response by Robot. At least 8 response | **Some IFTTT will base on previous voice response** |
|  |  |  |  |  |

## output structure

"output structure" contains the following fields:

**name**: (description of output actions)

**class**: (execute internal functions, Mechanics, LED, Zigbee Command, Cloud command, Voice Reply)

**subclass**:

**execute**: (execute commands, able to launch multiple programs)

**value**:

**priority**: (1-8, 1 being the highest priority, ***or follow "IFTTT structure"***)

**end\_condition**: (conditions to end output process)

**TBD1**:

**TBD2**:

The following is an example of output structures:

output(126)=struct( ...

'name', {{'push message'}}, ...

'class',{{''}}, ...

'subclass',{{''}}, ...

'execute',{{'Print(1234)'}}, ...

'value',{{'9'}}, ...

'priority',{{'10'}}, ...

'end\_condition',{{''}});

Since some functions are real time based, including sound transmission, voice-text recognition, text-voice machine reading, noise cancel, face detect and track, story and music play, music dance, etc. , the following table define work mode, and which functions are running in certain work mode.

|  |  |  |  |
| --- | --- | --- | --- |
| # | Work Mode Name | Description | Functions to run |
| 1 | Sleep Mode | Sleep only can wake up by press key, tap/touch robot or internet | 1)INT monitor by key press  2)Local Gsensor, Touch Sensor monitor  3)Network, WiFi |
| 2 | Standby Mode | Standby can wake by voice, on top of Sleep mode wake up method | 1)INT monitor by key press  2)Local Gsensor, Touch Sensor monitor  3)Network, WiFi  4) Local Voice-Text Recognition Engine |
| 3 | Voice Interaction  No-WiFi  [First Power Up, or WiFi lost] | After reset to factory condition, monitor WiFi join process. Or WiFi Lost | 1) Local Voice-Text Recognition Engine  2) Local Text-Voice Machine Reading  3) **Sound transmission monitor** and Config Network  4) Face detect and track  5) [Use IFTTT config. table:] play music and story, and launch noise cancel]  6) [Use IFTTT config. table:] take photo (store in SD card)  7) [Use IFTTT config. table:] take video (store in SD card)  8) [Use IFTTT to] manage Zigbee Devices (i.e. no push message) |
| 4 | Voice Interaction mode (with WiFi connection, no music play) | Robot WiFi properly configured. Voice Interaction with user | 1) Networking functions  2) Local Voice-Text Recognition Engine  3) Cloud Voice-Text Recognition Engine  4) [Use IFTTT to allow] Cloud reply to answers  5) Local Text-Voice Machine Reading  6) Face detect and track  7) [Use IFTTT to] take photo (store in SD card)  8) [Use IFTTT to] take video (store in SD card)  9) [Use IFTTT to] send photo to WeChat  10) [Use IFTTT to] manage Zigbee Devices  [ functions are important:  1) Voice Translation: "Mr. Moore, What is the English word for "猫“？" or "小墨，英语（英文）猫怎么说？“。 Reply: "Cat"]  2) Photo to send to WeChat, and mark "**Photo by robot Mr. Moore**" |
| 5 | Spelling Context Mode | In this mode, the robot gives user a set of English words, user spell each word, the robot check if spelling is correct. | 1) Local Voice-Text Recognition Engine  2) Local Text-Voice Machine Reading  3) Face detect and track  4) [Use IFTTT to] get word table (local)  [If WiFi connected:]  5) Network functions  6) English word from internet??  7) **Generate "Score Card" send to WeChat.**  8) [Use IFTTT to allow] Cloud Voice-Text Recognition Engine  9) [Use IFTTT to allow] Cloud reply to answers  10) [Use IFTTT to] manage Zigbee Devices |
| 6 | Music/Story Play Mode | Play music and still support all voice functions. | 1) Local Voice-Text Recognition Engine  2) [Use IFTTT to allow] Face detect and track  3) [Use IFTTT to] take photo (store in SD card)  4) [Use IFTTT to] take video (store in SD card)  5) [Use IFTTT to] send photo to WeChat  6) Select or random play music  7) Run noise cancel  8) [Use IFTTT to] manage Zigbee Devices  [Photo and Video taking should not disturb music playing] |
| 7 | Pattern Recognition Mode (with WiFi connected) | Robot Recognize Pattern, and read out | 1) Networking functions  2) Local Voice-Text Recognition Engine  3) Cloud Voice-Text Recognition Engine  4) Local Text-Voice Machine Reading  5) [Use IFTTT to allow] Face detect and track  6) [Use IFTTT to allow] take photo (store in SD card)  7) Baidu Pattern Recognition  8) [Use IFTTT to] manage Zigbee Devices |
| 8 | Music Dance Mode | Listen to music and dance. (Music is either from robot play or other devices ) | 1) Local Voice-Text Recognition Engine  2) Local Text-Voice Machine Reading  3) [Use IFTTT to] Face detect and track  [If music play by robot:]  4) Select or random play music  5) Run noise cancel (sound source from PCM?)  6) [Use IFTTT to] manage Zigbee Devices |
| 9 | IPCam Mode | Standard IPCam function | 1) Smart Bridge Program  2) [Use IFTTT to] manage Zigbee Devices(???) |
| 10 | Test and Manufacture Mode |  |  |
| 11 | Firmware Update mode | Download new firmware, suspend all non-related functions | 1) Network functions to "Pilot Labs" server |

Output structures is 13bit address, with [0:4095]. The following table defines input class and subclass:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | class | subclass | description | Special note |
| 1 | Work Mode (16) | sleep(1) |  |  |
| Standby(1) |  |  |
| Voice Interaction  No WiFi (1) | IFTTT to define optional features |  |
| Voice Interaction  With WiFi (1) | IFTTT to define optional features |  |
| Spelling Contest(1) | IFTTT to define optional features |  |
| Music/Story Play(1) | IFTTT to define optional features |  |
| Pattern Recognition(1) | IFTTT to define optional features |  |
| Music Dance (1) | IFTTT to define optional features |  |
| IPCam(1) |  |  |
| Test and Manufacture(1) |  |  |
| Firmware Update(1) |  |  |
|  |  |  |  |  |
| 2 | Execute command () | Firmware(4)  [0:3] | update Firmware version related | i.e. update firmware version. |
| Audio in(4) | Execute audio record related functions | i.e. run noise cancel. |
| Audio out(4) | Execute audio play related functions |  |
| Camera(4) | Camera related functions | i.e. open camera, take photo, run face detect |
| Power(4) | Change power mode |  |
| WiFi(16) | WiFi related functions |  |
| Network(16) | Network related functions |  |
| Time(16) | System time related functions |  |
| Other() |  |  |
| 3 | mechanics(128) | static(64) | Turn head, eye, eye lid to a position |  |
| dynamic(64) | Eye blink, face tracking, music |  |
|  |  |  |  |  |
| 4 | Zigbee Co(16)  [768:783] | TBD |  |  |
| TBD |  |  |
|  |  |  |  |  |
| 5 | Zigbee Device(192) |  |  |  |
|  |  |  |
| Command(192) | Control zigbee device |  |
|  |  |  |  |  |
| 6 | SD card(8) | content (8)  [976:979] | Change content on SD card | i.e. delete, write, copy, etc. |
|  |  |  |
|  |  |  |  |  |
| 7 | LED |  |  |  |
|  |  |  |  |  |
| 8 | Execute Cloud functions(64)  [not include voice] | WeChat(16) |  |  |
| Face Recognition(16) |  |  |
| Pilot Labs cloud(16) |  |  |
| TUTK(16) |  |  |
|  |  |  |  |  |
| 9 | Voice Reply() | Local stored reply (1024) | Pre-stored reply when user voice matched |  |
| Expression(256) | Pre-stored expression when user voice matched | i.e. laugh |
| Sound(256) | Play sound i.e. water drop, hammer, etc. |  |
| Cloud voice reply (2088) | Filter cloud reply |  |
|  |  |  |  |  |

## IFTTT table

"IFTTT mapping table" is defined to configure how system responds to each input. The structure contains the following fields:

**name**: (description of IFTTT actions)

**class**: (system, WiFi, Network, Zigbee Co, Zigbee Device, SD card, Voice, Power, Time, GIS, Cloud Command, IFTTT, Local Sensor, User Information)

**subclass**:

**condition**: (logic condition for "IF THIS THEN THAT")

**output**: (index of output structure)

**priority**: (1-8, 1 being the highest priority)

**nextaction**: (define execute more function after output is finished)

**TBD1**:

**TBD2**:

IFTTT(576)=struct( ...

'name', {{'3 conditions trigger sample output'}}, ...

'class',{{'voice'}}, ...

'subclass',{{'INT Voice Matched'}}, ...

'condition', {{'i(4356) & (status(101).value==''01-Nov-2015'' |status(102).value==1) | 1 & 1 & 1'}}, ...

'output',{{'126' '127'}}, ...

'priority',{{'1'}}, ...

'nextaction',{{'output(128)'}}, ... %when finish required output, goes another output or IFTTT

'TBD1',{{''}}, ...

'TBD2',{{''}});

## matlab sample

the following is a matlab sample of how system works:

clear all;

% This program is an example of Input, System Status, IFTTT config, and

% Output

%% IFTTT struct: field: name condition output priority TBD1 TBD2

% IFTTT(576): On Monday if i4356 then trigger output(126)

IFTTT(576)=struct( ...

'name', {{'3 conditions trigger sample output'}}, ...

'class',{{'voice'}}, ...

'subclass',{{'INT Voice Matched'}}, ...

'condition', {{'i(4356) & (status(101).value==''01-Nov-2015'' |status(102).value==1) | 1 & 1 & 1'}}, ...

'output',{{'126' '127'}}, ...

'priority',{{'1'}}, ...

'nextaction',{{'output(128)'}}, ... %when finish required output, goes another output or IFTTT

'TBD1',{{''}}, ...

'TBD2',{{''}});

IFTTT(999)=struct('name', {{''}},'class',{{''}},'subclass',{{''}},'condition', {{''}}, 'output',{{''}},'priority',{{''}},'nextaction',{{''}},'TBD1',{{''}},'TBD2',{{''}});

%% input struct: name class subclass static\_or\_dynamic priority TBD1 TBD2

% Use "Time & Date" as example

input(4356)=struct( ...

'name', {{'INT Voice Matched: "good moorning"'}}, ...

'class',{{'Voice'}}, ...

'subclass',{{'INT Voice Matched'}}, ...

'static\_or\_dynamic',{{'dynamic'}}, ...

'priority',{{'1'}}, ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

input(9999)=struct('name', {{''}},'class',{{''}},'subclass',{{''}},'static\_or\_dynamic',{{''}},'priority',{{''}},'TBD1',{{''}},'TBD2',{{''}});

%% status struct: name class subclass value

% system date and time as example

status(101)=struct( ...

'name', {{'system date'}}, ...

'class',{{'time'}}, ...

'subclass',{{'local'}}, ...

'value',[date], ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

status(102)=struct('name', {{'system weekdate'}},'class',{{'time'}},'subclass',{{'local'}},'value',[weekday(date)],'TBD1',{{''}},'TBD2',{{''}});

fixclock=fix(clock);

status(103)=struct('name', {{'system now'}},'class',{{'time'}},'subclass',{{'local'}},'value',[fixclock(4:6)],'TBD1',{{''}},'TBD2',{{''}});

%% output struct: name class subclass execute value priority end\_condition

output(126)=struct( ...

'name', {{'push message'}}, ...

'class',{{''}}, ...

'subclass',{{''}}, ...

'execute',{{'Print(1234)'}}, ...

'value',{{'9'}}, ...

'priority',{{'10'}}, ...

'end\_condition',{{}}, ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

output(127)=struct( ...

'name', {{'sound alert: siren'}}, ...

'class',{{''}}, ...

'subclass',{{''}}, ...

'execute',{{'Print(5678)'}}, ...

'value',{{'9'}}, ...

'priority',{{'10'}}, ...

'end\_condition',{{''}}, ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

output(128)=struct( ...

'name', {{'camera cntl: take photo'}}, ...

'class',{{''}}, ...

'subclass',{{''}}, ...

'execute',{{'Print(abcd)'}}, ...

'value',{{'9'}}, ...

'priority',{{'10'}}, ...

'end\_condition',{{''}}, ...

'TBD1',{{''}}, ...

'TBD2',{{''}});

%% main function

logfile='main.log';

fid=fopen(logfile,'w');

%%% recognize "good morning"

Voice\_recognized='good morning';

System\_INT(1)=struct('name', {{'INT Voice Matched: "good moorning"'}},'class',{{'INT'}},'subclass',{{'INT Voice Matched'}},'static\_or\_dynamic',{{'dynamic'}},'priority',{{'1'}});

% search system input pool

l=1;

for n=1:9999

if size(char(input(n).subclass))==size(char(System\_INT(1).subclass))

if char(input(n).subclass)==char(System\_INT(1).subclass)

i(n)=1;

valid\_input(l)=n; l=l+1;

end

end

end

%% generat log

fprintf(fid, 'valid Input Found:');

fprintf(fid, '%i \n', n);

%% find if this input is related to IFTTT settings

for k=1:999

IFTTT\_condition\_string=char(IFTTT(k).condition);

valid\_IFTTT=size(char(IFTTT\_condition\_string));

if valid\_IFTTT(2)>=6

str2num(IFTTT\_condition\_string(3:6))

if i(str2num(IFTTT\_condition\_string(3:6)))==1

nIFTTT\_to\_execute=k

% generate log

fprintf(fid, 'Related IFTTT Found:');

fprintf(fid, '%i \n', k);

% if IFTTT condition met, execute

if eval(IFTTT\_condition\_string)==1

output\_finished=0; % indicator to monitor if output actions are all finished

Noutput=str2num(char(IFTTT(k).output));

output\_action\_number=size(Noutput); %% multiple actions required

for g=1:output\_action\_number

output(Noutput(g))

end

% generate log

fprintf(fid, 'Output Executed:');

fprintf(fid, '%i \n', Noutput);

output\_finished=1;

% see if more action should take

str\_nextaction=char(IFTTT(k).nextaction);

if str\_nextaction(1:6)=='output'

outputpointer=str2num(str\_nextaction(8:10));

output(outputpointer)

end

if str\_nextaction(1:5)=='IFTTT'

IFTTTpointer=str2num(str\_nextaction(7:9));

%execute another IFTTTT

end

% clear INT

clear i(valid\_input(:))

end

end

end

end

fclose(fid);