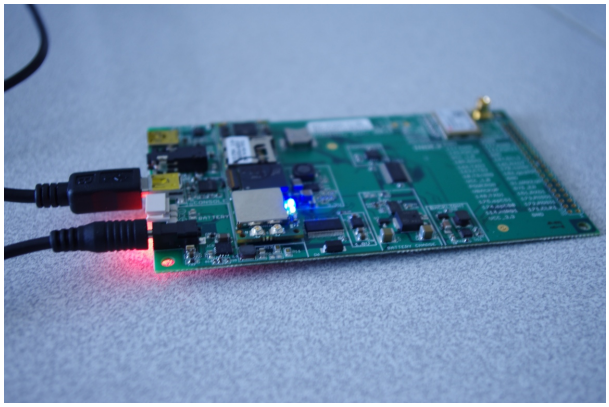
Développement sur Gumstix

Master ISIM-SIC

Pierre Andry
Université Cergy-Pontoise
andry@ensea.fr

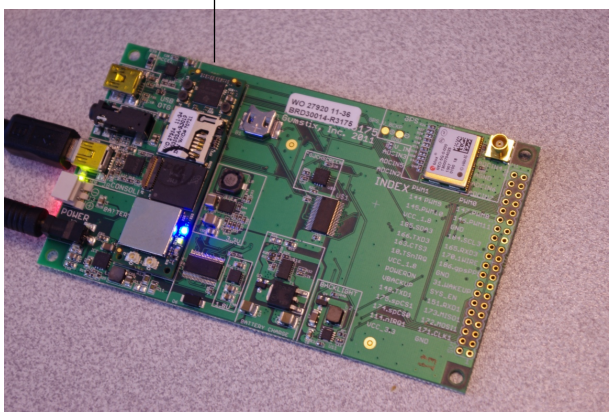
Gumstix

Principe d'un pc embarqué de dernière génération



- Overo AIR
- Carte d'extension : Gallop43
- Linux
- quelques centimètres carrés

Gumstix



Processor: ARM Cortex-A8 CPU 600 MHz

Memory: 512MB RAM 512MB Flash

Features:
OMAP3503 Application Processor **802.11b/g** wireless communications
Bluetooth communications
microSD card slot
TPS65950 Power Management

Expandability:
via one 140-pin expansion board of Gumstix Overo series or custom,
140-pin expansion board
via 27-pin camera board
Connections:
- (2) 70-pin connectors with 140 signals for:
- I2C, PWM lines (6), A/D (6), 1-wire
- UART, SPI, Extra MMC lines
- Headset, Microphone
- Backup battery
- High Speed USB Host and USB OTG

(1) 27-pin connector with signals for camera board
(2) x u.fl antenna connectors
Power:
Powered via expansion board (Overo series or custom) connected to dual 70-pin connector
Layout:
For OEM design, use layout version R2606 or later, shown here.
Size:
17mm x 58mm x 4.2mm
(0.67 in. x 2.28 in. 0.16 in.)
Weight:
GUM3503A @ 42.6g (incl. shipping case & 2 x antenna)
GS3503A @ 5.6g

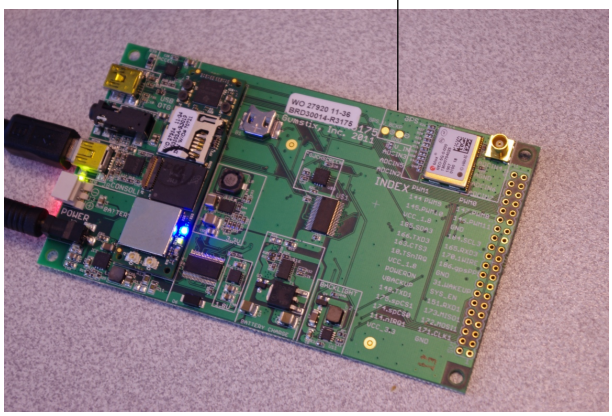
Temperatures:
Built with components rated 0C < T <75C
Mounting:
Four (4) x #0 mounting holes for securing to Overo-series, or custom, expansion board

Software

Linux Angström

- Projet basé sur open embeded
- Linux pouvant tourner sur 4MB de mémoire flash
- embarqué sur beagle board
- la compilation croisée passe par gcc-arm-linux-gnueabi
- *(le gumstix autorise la compilation native)*

Gallop43



Features:
GPS via the u-blox NEO-5G module
3-axis accelerometer

LCD-ready for 4.3" touch-screen
LCD display @ 24 bits per pixel
Touchscreen controller

USB mini-AB with OTG signals
USB mini-B console port

Coin cell backup battery holder
3.5-mm stereo headset jack

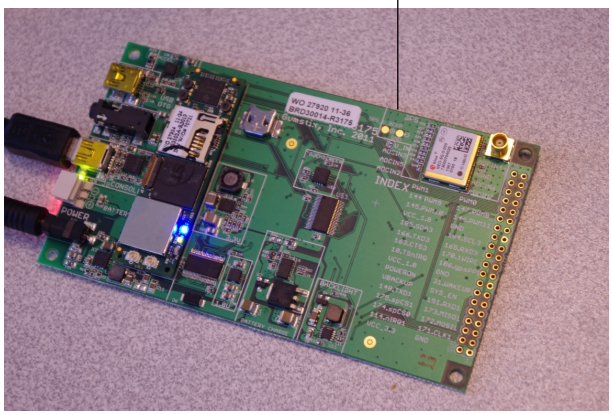
MCX antenna connector for GPS
Two (2) user-configurable LEDs
Two (2) user-configurable push buttons
Forty (40) signals available on 0.100" through-holes at 1.8V logic
Two (2) two-wire serial ports
One 1-wire port
6 PWM lines
I2C port
SPI bus
6 A/D input lines
processor control signals

Power:
Connect a 5v wall adapter to power this expansion board, the connected Overo COM and the attached LCD touch screen
Alternate power jack provided for 2-cell NiMH battery pack

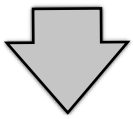
Connecting both the 5v wall adapter and a MiMH battery pack will charge the NiMH batteries
Connections:
2 x 70-pin connectors for connection of any Overo COM

Computing:
Requires any Gumstix Overo COM
Temperatures:
Built with components rated -20< T <70C
Size:
118.2mm x 67.2mm
Weight:
BRD30014 @ 36.2g

Projet



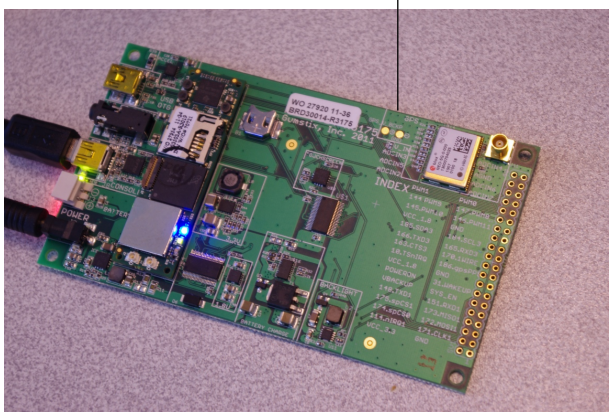
- Récupérer les informations des accéléromètres
- calculer la dérive -> la vitesse et la position lors de mouvements simples de translation sur le table.
- les envoyer périodiquement à une station distante via WIFI (UDP)



- récupérer les informations via UDP
- *eventuellement* afficher les info en 3D

Projet

- A faire en binomes,
- bien découper le temps de travail et l'accès aux gumstix
- bien cerner le tâches a effectuer



Connexion

1. Ne pas **brancher** ni **connecter** la carte
2. Vérifiez que minicom est installé
3. Vérifiez que le fichier .minirc.dfl existe dans votre HOME
4. vérifiez que minicom essaye bien d'ouvrir /dev/ttyUSB0
5. connecter la carte sur le port USB
6. lancez minicom en 115200 8N1
7. branchez la carte

To **configure Minicom :**

1. run `minicom`
2. 'Ctrl-A' 'O' -> "Serial Port Setup" choose the following
   - Serial Device: **/dev/ttyUSB0**
   - Bps/Par/Bits: **115200 8N1**
   - Hardware Flow Control: **No** (this is important)
   - Software Flow Control: **No**
3. return to the main configuration menu
4. save this setup
5. Exit

Plug the power adapter into the power jack of the Gumstix. When connected and powered, you should see a message from U-boot followed by the normal Gumstix boot sequence in the minicom window.

Log in for the first time with username *root* and password *gumstix*

When finished, you can exit Minicom by typing **[CTRL-A]** then pressing **Q**.
This exits without running the normal modem reset sequence, which will only send garbage to the Gumstix.

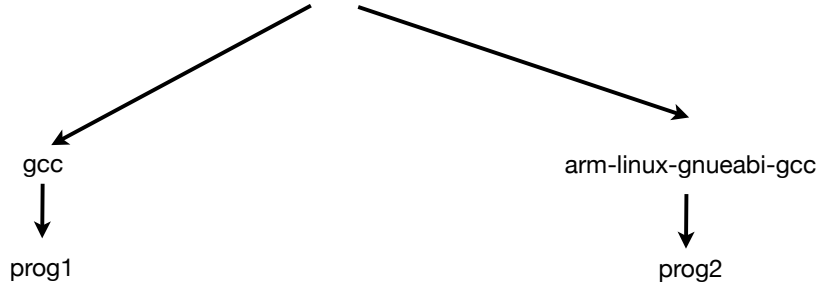
Software

Linux Angström

Bien identifier la machine cible de la machine hote

cross compilateur : gcc-arm-linux-gnueabi

compilez le même code source «hello world» sur la **machine hôte**



utilisez la commande file sur prog1, puis prog2, que nous indique t'elle ?

créez un makefile qui permet de choisir la la variable CC, testez

Transfert de fichier

communication gumstix <-> pc hote (1) : par lien WIFI ad-hoc

GUMSTIX

ESSID GMSTX

- channel 2
- no encryption
- mode ad-hoc
- 192.168.10.10
- utiliser ifconfig, iwconfig, ping

PC HOTE

ESSID GMSTX

- channel 2
- no encryption
- mode ad-hoc
- 192.168.10.2
- utiliser ifconfig, iwconfig, ping

iwconfig wlan0 power on mode Ad-Hoc essid "GMSTX" channel 2

Une fois la connection établie, utiliser scp pour le transfert de fichier

Transfert de fichier

communication gumstix <-> pc hote (1) : par lien WIFI ad-hoc

REMARQUES

Souvent il faut enlever network manager :

```
opkg remove networkmanager   (machine cible)
sudo service network-manager stop (machine hôte)
```

(car souvent network-manager tentera de rebasculer la connection wifi sur d'autres signaux)

Pour éviter les plantages, il faut retirer wlan0 des interfaces avant de la configurer :

```
ifconfig wlan0 down
iwconfig wlan0 blablablah/paramètres de configuration wifi
ifconfig wlan0 up
ifconfig wlan0 moniP
```

Pour scanner les réseaux WIFI à partir de votre carte wifi :

```
iwlist wlan0 scan
```

Transfert de fichier

se connecter sur le gumstix depuis la machine hote sur le gumstix

```
ssh root@192.168.2.10
```

```
scp fichiermachinehote root@192.168.2.10:~
```

Transfert de fichier

communication gumstix -> pc hote (2) : par lien série RSD32 (protocole zmodem)

Cross-compilez pour Gumstix : lrzsz (utilitaire d'envoi et de réception des fichier via le port série) et installer lrzsz sur le Gumstix

GUMSTIX

- lancer rz

PC HOTE

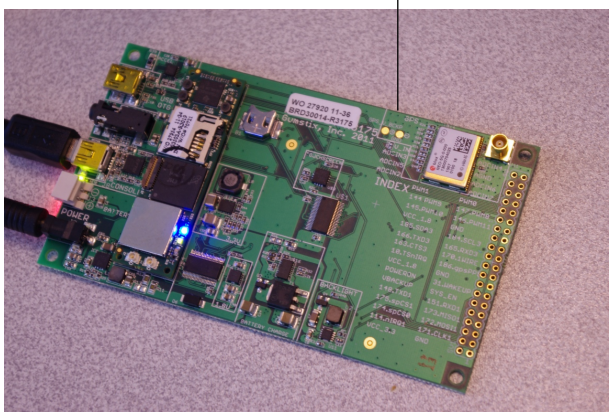
- lancer minicom
- se logger sur le gm
- [Ctrl A] - S
- transferer le fichier en zmodem

Transfert de fichier

par mémoire carte micro sd

- /media/mmcblk0p1

Projet

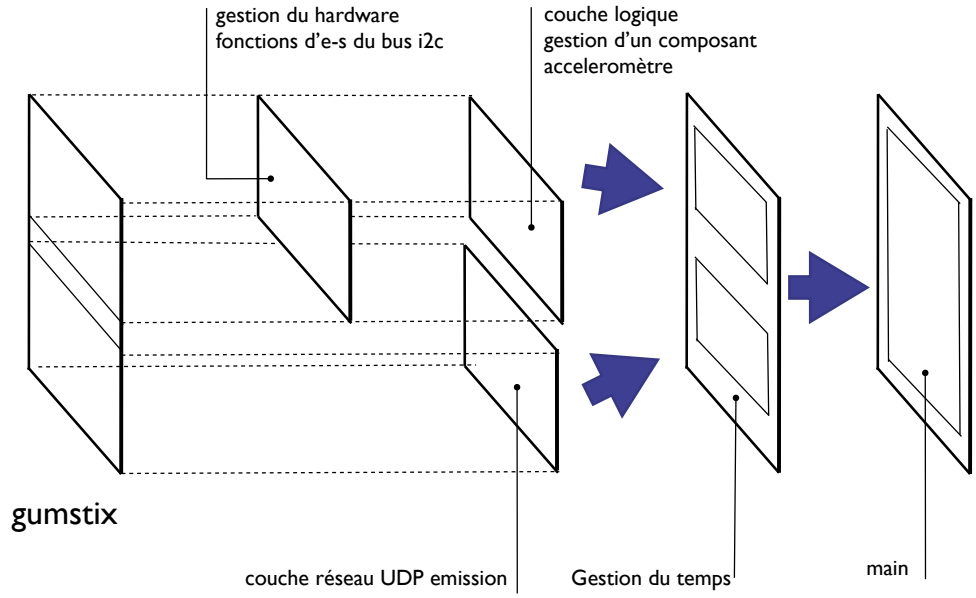


- Récupérer les informations des accéléromètres
- calculer la dérive -> la vitesse et la position lors de mouvements simples de translation sur le table.
- les envoyer périodiquement à une station distante via WIFI (UDP)

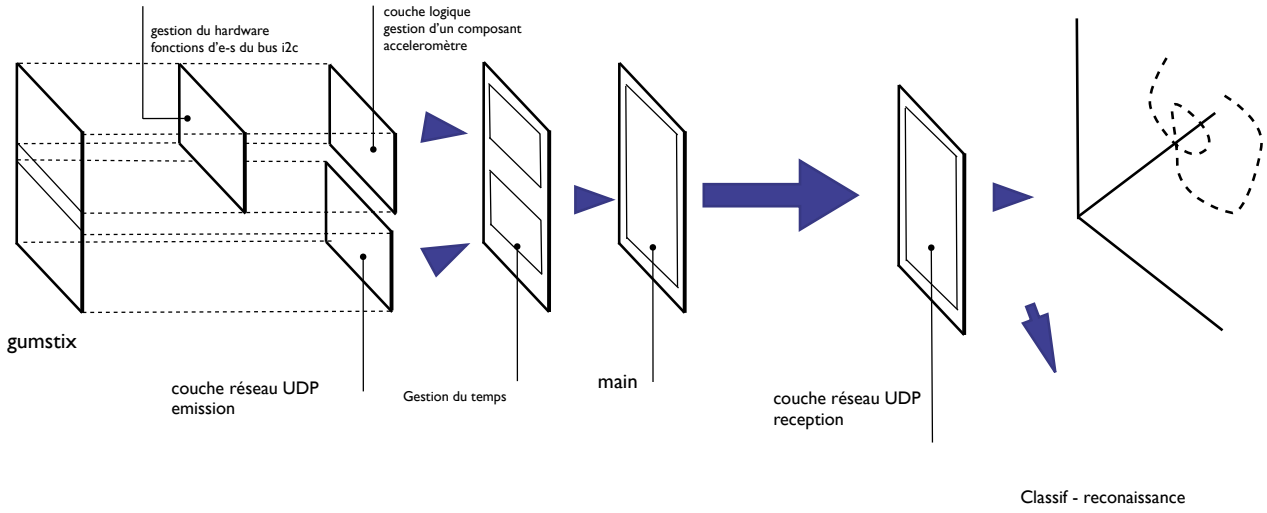
- récupérer les informations via UDP
- *eventuellement* afficher les info en 3D

Vue d'ensemble

Application de reconnaissance de gestes de type "wiimote"



Vue d'ensemble



Traitement des Accéléromètres

L'accélération d'un objet est la dérivée de sa vitesse
La vitesse d'un objet est la dérivée de sa position

$$\vec{a} = \frac{d\vec{v}}{dt} \text{ and } \vec{v} = \frac{d\vec{s}}{dt} \therefore \vec{a} = \frac{d(d\vec{s})}{dt^2}$$

A l'inverse, il faut intégrer 2 fois l'accélération pour retrouver la position d 'un objet

$$v = \int(\vec{a})dt \text{ and } \vec{s} = \int(\vec{v})dt \therefore \int\left(\int(\vec{a})dt\right)dt$$

source : freescale semi-conductor application note, an 3397

Traitement des Accéléromètres

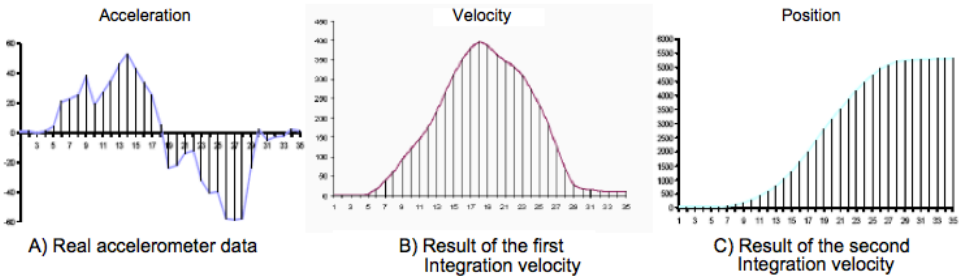


Figure 6. Proportional Approximation of the Instantaneous Position

source : freescale semi-conductor application note, an 3397

Traitement des Accéléromètres

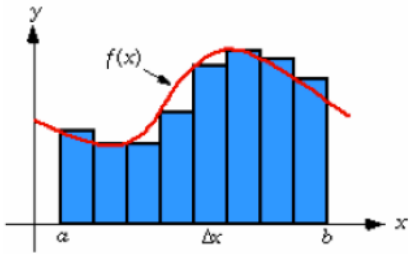


Figure 1. Sampled Accelerometer's Signal

With the previous concept about "areas below the curve" a deduction can be made: Sampling a signal gets us instant values of its magnitude, so small areas can be created between two samples. In order to create a coherent value, sampling time must always be the same. Sampling time represents the base of this area while the sampled value represents its height. In order to eliminate multiplications with fractions (microseconds or milliseconds) involving floating points in the calculation we assume the time is a unit.

$$\int_a^b f(x)dx = \lim_{n \to \infty} \sum_{i=1}^{n} f(x_i)\Delta x$$

Where:

$$\Delta x = \frac{b-a}{n}$$

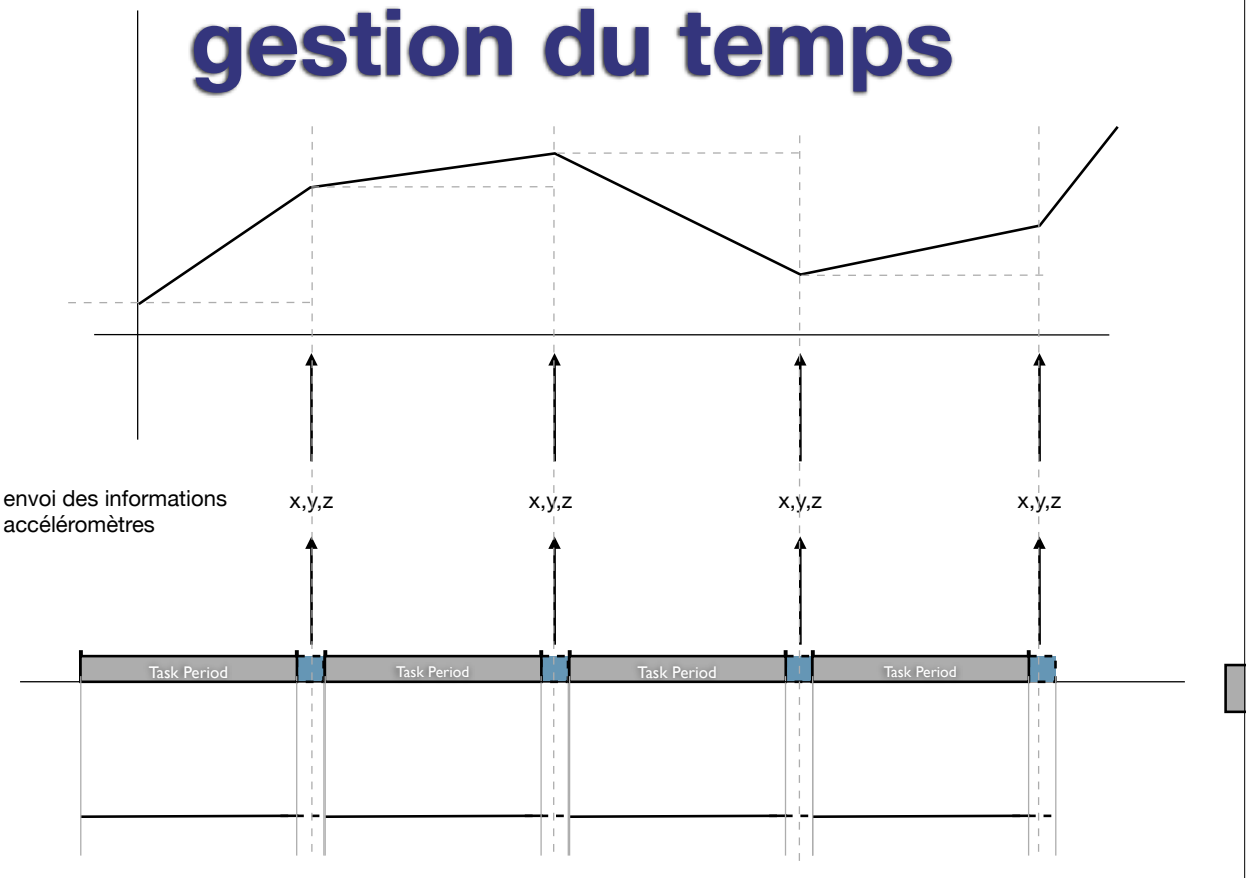
source : freescale semi-conductor application note, an 3397

mesure du temps

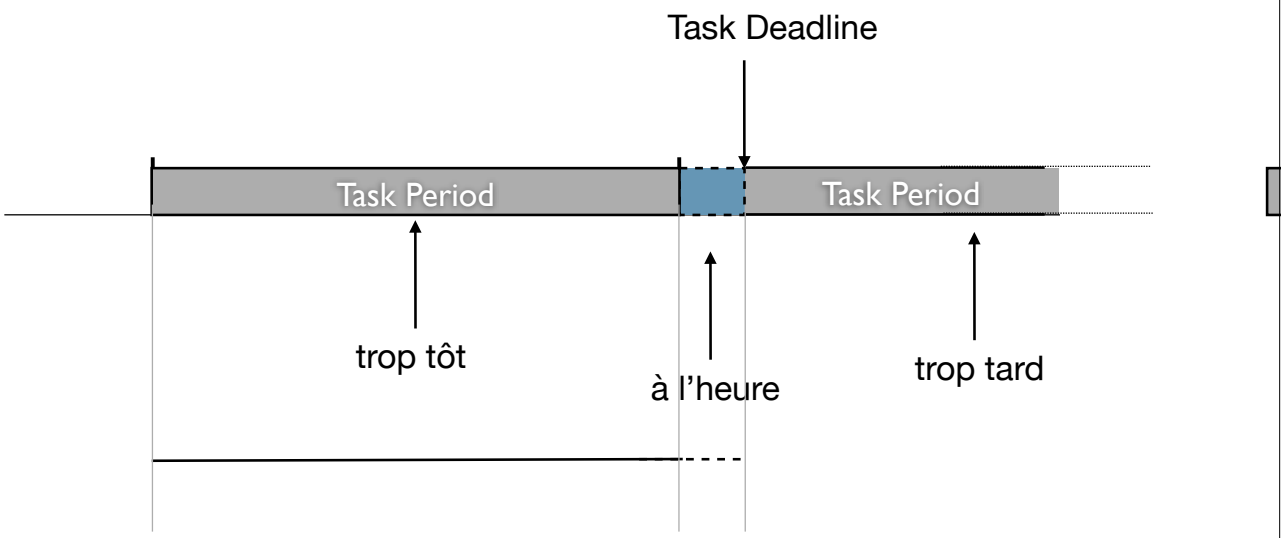
Nous allons récupérer des information numériques (accélérations) que nous allons traiter pour reconstruire une information de position.

l'accélération dépends du temps, il nous faut donc des intervalles **réguliers** de mesure, sinon notre base de calcul de reconstruction

gestion du temps



gestion du temps



mesure du temps

```
  /*initializing the time reference*/
  gettimeofday(&start, 0);

 /*main loop */
  while (1) {

    /*mesuring time : checkpoint */
    gettimeofday(&checkpoint, 0);
    diff=(checkpoint.tv_sec-start.tv_sec) * 1000000L + (checkpoint.tv_usec-start.tv_usec); /* calculate time elapsed */

      gettimeofday(&start, 0); /* re-init the time reference for next calculation*/
#ifdef __GTIMETRACE
     printf("temps écoulé=%lld usec\n",diff);
#endif
}
```

- checkpoint et start sont des variables de type : struct timeval
- struct timeval {

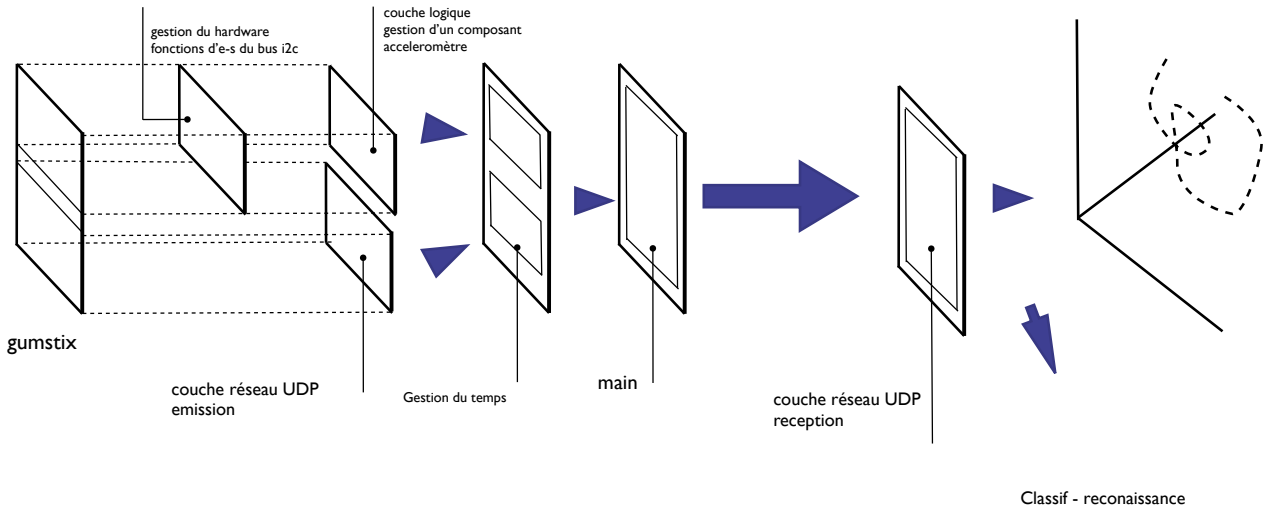
  time_t tv_sec; /* seconds since Jan. 1, 1970 */

  suseconds_t tv_usec; /* and microseconds */

gestion du temps

```
/*initializing the time reference*/
gettimeofday(&start, 0);

/*main loop */
while (1) {

  /*mesuring time : checkpoint */
  gettimeofday(&checkpoint, 0);
  diff=(checkpoint.tv_sec-start.tv_sec) * 1000000L + (checkpoint.tv_usec-start.tv_usec); /* calculate time elapsed */

  if (diff < TASK_PERIOD ) ; /* we are toot early : do nothing*/
  else {
  gettimeofday(&start, 0); /* re-init the time reference for next calculation*/
#ifdef __GTIMETRACE
   printf("temps écoulé=%lld usec\n",diff);
#endif
   if (diff > TASK_PERIOD + TASK_DEADLINE) fprintf (stderr,"***echeance manquée \n"); /* we are too late*/
   else {  /*we are on time : do real-time stuff here*/

    /* collecting datas here & Arya is the bete !!*/

    /*sending datas here*/

   }

  }

}
```

Vue d'ensemble



simulation envoi-réception

code du client

```
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>

#define BUFLEN 512
#define PORT 9930


void diep(char *s)
{
  perror(s);
  exit(1);
}

  #define SRV_IP "127.0.0.1"
```

```
int main(void)
 {

    struct sockaddr_in si_other;
    int s, slen=sizeof(si_other);
    char buf[BUFLEN];
    int val1,val2,val3;
    int tmp = 0;

    val1=val2=val3=4;  //simule 3 valeurs accelerometres

        if ((s=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP))==-1)
          diep("socket");

      memset((char *) &si_other, 0, sizeof(si_other));
      si_other.sin_family = AF_INET;
      si_other.sin_port = htons(PORT);

      if (inet_aton(SRV_IP, &si_other.sin_addr)==0) {
        fprintf(stderr, "inet_aton() failed\n");
        exit(1);
       }

        while (1) {

                /*envoi des informations*/
                sprintf(buf, "%d %d %d\n", val1,val2,val3);
                if (sendto(s, buf, BUFLEN, 0, &si_other, slen)==-1) diep("sendto()");

                tmp ++;
                val1=val2=val3=tmp % 254;     /* changement des informations (simul*/

              }

        close(s);
        return 0;
 }
```

simulation envoi-réception

code du serveur

```
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

#define BUFLEN 512
#define PORT 9930

      void diep(char *s)
      {
        perror(s);
        exit(1);
       }
```

```
   int main(void)
   {
       struct sockaddr_in si_me, si_other;
       int s, i, slen=sizeof(si_other);
       char buf[BUFLEN];

       if ((s=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP))==-1)
         diep("socket");

       memset((char *) &si_me, 0, sizeof(si_me));
     si_me.sin_family = AF_INET;
     si_me.sin_port = htons(PORT);
     si_me.sin_addr.s_addr = htonl(INADDR_ANY);
     if (bind(s, &si_me, sizeof(si_me))==-1)
          diep("bind");

      while (1) {
        if (recvfrom(s, buf, BUFLEN, 0, &si_other, &slen)==-1)
              diep("recvfrom()");
        printf("Received packet from %s:%d\nData: %s\n\n",
                                  inet_ntoa(si_other.sin_addr),
ntohs(si_other.sin_port), buf);
       }

       close(s);
     return 0;
   }
```

simulation envoi-reception

Testez sur station classique la communication client - serveur

Ajouter la mesure du temps

Ajoutez le contrôle du temps réel mou : le temps mesuré sert à définir si l'on est trop tôt ou trop tard et à cadencer l'envoi des messages.

réglez pour un envoi toutes les 50ms +ou- 10ms

Notez s'il y a des échéances manquées

simulation envoi-reception

Testez sur station classique la communication client - serveur

Ajouter la mesure du temps

Ajoutez le controle du temps réel mou : le temps mesuré sert à définir si l'on est trop tôt ou trop tard et a cadencer l'envoi des messages.

reglez sur un envoi toutes le 50ms +ou- 10ms

Notez s'il y a des échéances manquées

Laissez reposer, et passez a la gestion des accéléromètres sur gumstix

gestion des accéléromètres

Accéléromètre 3 axes.

L'acceleromètre est un capteur LIS3DE dont le datasheet est dans le répertoire ci-joint. lis3de.pdf.

-> Page 22 se trouve le registre mapping.
on note que les infos x, y et z se trouvent aux registres respectifs : 29, 2B,et 2D

-> Page 23 contient des informations sur le contrôle des données (registre 20) , qui peut être écrit pour modifier la fréquence de mise a jour des données. notamment le champ DR, avec 0 correspondant à 100HZ.
Par défaut, on a 0000111 avec les 3 derniers bits a 1 pour lire les infos sur les infos en x,y et z.

gestion des accéléromètres

Gestion des accéléromètres

il faut d'abord activer l'autorisation en lecture de l'accéléromètre :

i2cset -y 3 0x1d 0x20 0x47 b

- Adresse du bus
- Adresse du capteur
- Adresse du registre
- Valeur

gestion des accéléromètres

Gestion des accéléromètres

Puis interroger les registres

en x :
i2cget -f -y 3 0x1d 0x29
en y :
i2cget -f -y 3 0x1d 0x2B
en z :
i2cget -f -y 3 0x1d 0x2D

- Adresse du bus
- Adresse du capteur
- Adresse du registre a lire

gestion des accéléromètres

Gestion des accéléromètres

- i2cset.c
- i2c-dev.h

•Reprendre la configuration écriture/lecture en C

•Autoriser la lecture des accéléromètres

•récupérer les données des accéléromètres

- i2cget.c
- i2c-dev.h
- busses.h
- utils.h

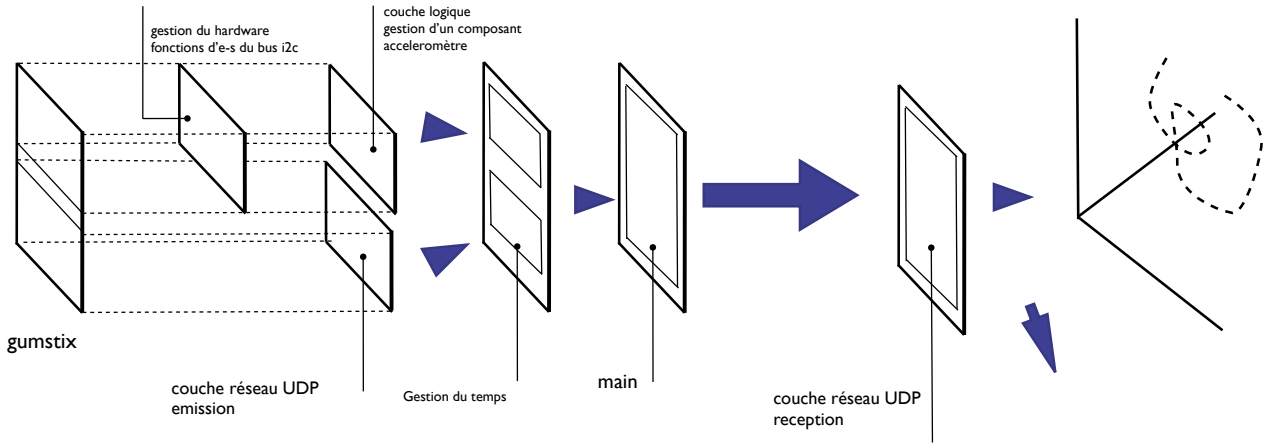
•récupérer les fichier sur machine “classique”

•afficher les valeurs des accéléromètres

- connexion TCP ou UDP réseau en C

http://lm-sensors.org/svn/i2c-tools/tags/V3-0-1/tools/

Vue d'ensemble



Traitement des Accéléromètres

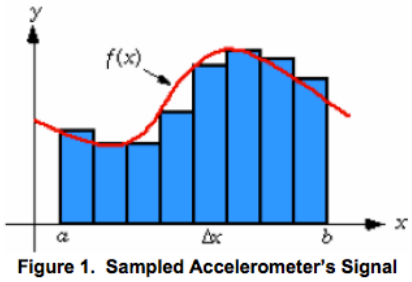


Figure 1. Sampled Accelerometer's Signal

With the previous concept about "areas below the curve" a deduction can be made: Sampling a signal gets us instant values of its magnitude, so small areas can be created between two samples. In order to create a coherent value, sampling time must always be the same. Sampling time represents the base of this area while the sampled value represents its height. In order to eliminate multiplications with fractions (microseconds or milliseconds) involving floating points in the calculation we assume the time is a unit.

$$\int_a^b f(x)dx = \lim_{n \to \infty} \sum_{i=1}^{n} f(x_i)\Delta x$$

Where:

$$\Delta x = \frac{b-a}{n}$$

source : freescale semi-conductor application note, an 3397

Traitement des Accéléromètres

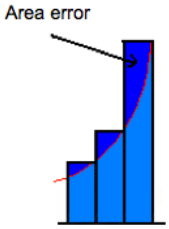


Figure 2. Errors Generated During Integration

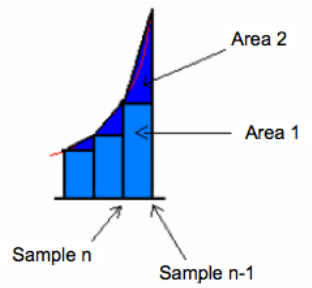


Figure 3. Errors of Integration Are Reduced with a First Order Approximation (Trapezoidal Method)

$$Area_n = Sample_n + \frac{|Sample_n - Sample_{n-1}|}{2} \times T$$

source : freescale semi-conductor application note, an 3397

Traitement des Accéléromètres

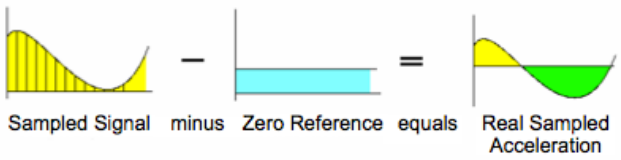
SOFTWARE DESIGN CONSIDERATIONS

The following steps and recommendations should be considered when implementing this kind of algorithm in a "real world" implementation.

- The signal is not noise free so it must be digitally filtered. The filter used in this algorithm is a moving average; the value to be processed is the result of averaging a certain amount of samples.
- Even with the previous filtering some data can be erroneous due to the "mechanical" noise, so another filter must be implemented. Depending on the number of samples filtered, a window of "real acceleration" can be selected (typically ± 2 sample steps for an average of 16 samples).
- A "no movement" state is critical to obtain correct data. A calibration routine is needed at the beginning of the application. This calibration value must be as accurate as possible.
- The real value of the acceleration is the sample minus the calibration value; it can be either positive or negative. This must never be ignored when declaring variables (signed).
- A faster sampling frequency implies more accurate results due the fact that error is reduced; yet more memory, timing, and hardware considerations are needed.
- The time between samples MUST always be the same. Errors can be generated if this time is not equal.
- A linear approximation between samples (interpolation) is recommended for more accurate results.

source : freescale semi-conductor application note, an 3397

Calibration



source : freescale semi-conductor application note, an 3397

Traitement du bruit

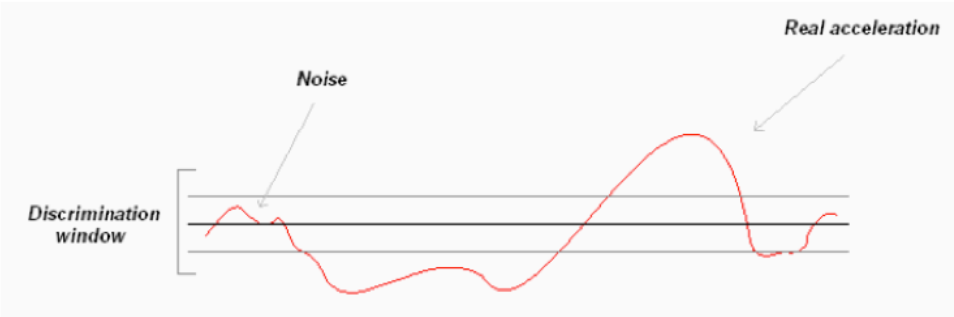


Figure 7. Software Discrimination Window To Reduce Mechanical Noise Effects

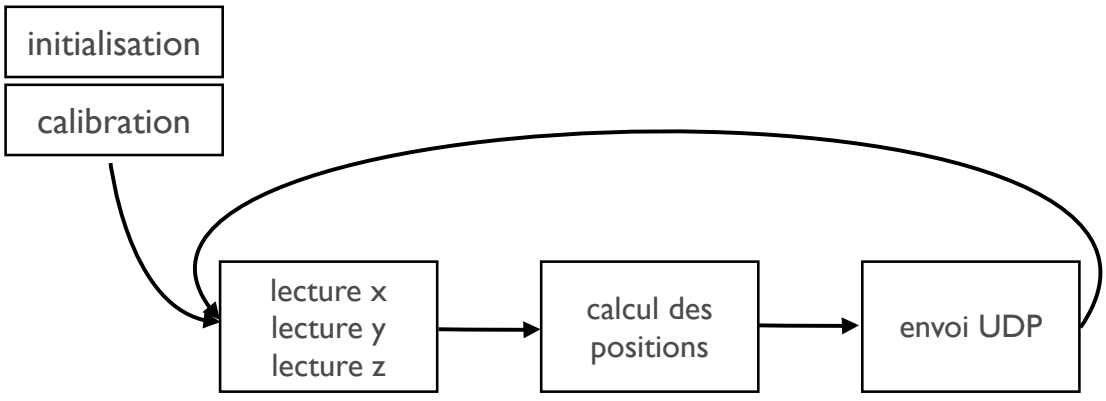
source : freescale semi-conductor application note, an 3397

notes sur la carte SD

montée sur le gumstix sous :
/media/mmcblk0p1

source : freescale semi-conductor application note, an 3397

Architecture souhaitée



source : freescale semi-conductor application note, an 3397