Pràctica 2

Accés a informació en el hardware

Salvador Llobet, Ignacio

Soria Magraner, Enrique

1. Accés al sistema

Per tal d'accedir a les dades del sistema, simplement hem tingut que incloure la capçalera "*system.h*" de *libnds*, i després accedir a l'estructura *PersonalData* des d'on poden extraure el nom d'usuari, aniversari, color del tema, etc. Després simplement hi ha que mostrar eixes dades en pantalla.

Imatge 1: Sortida del programa

2. Funcions relacionades amb el pas del temps

A aquest apartat ens donen tres funcions, de tres exemples, i ens pregunten sobre elles o els seus arguments. Passem a veure les funcions i resoldre les preguntes pas a pas.

```
timerStart(0, ClockDivider_1024, 0, NULL);
```

Aquesta funció inicialitza un temporitzador que cridarà al seu últim argument (en aquest cas no) segons la freqüència que se li indique. El segon argument indica el canal del temporitzador, mentre

que el tercer són el nombre de ticks que deuen passar fins que es produïsca l'overflow.

```
timerStart(0, ClockDivider_1024, TIMER_FREQ_1024(5), timerCallBack);
```

En aquest cas la novetat són els dos últims arguments: el tercer indica que es cridarà al quart una vegada cada cinc segons. El quart, com ja hem dit, serà la funció a la que cridarà el timer, en aquest cas, cada cinc segons.

Per acabar se'ns pregunta sobre l'exemple *RealTimeClock*. En concret, quines són les funcions que es gasten per endevinar l'hora i la data.

Si mirem el codi, podem observar que aquestes són les línies concretes on s'extrau la informació del sistema:

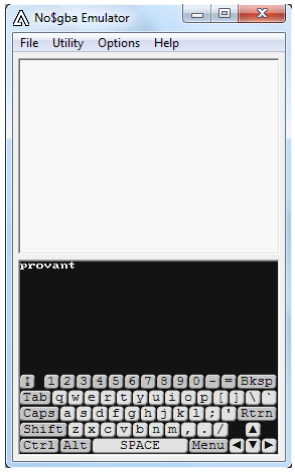
```
time_t unixTime = time(NULL);
struct tm* timeStruct = gmtime((const time_t *)&unixTime);

hours = timeStruct->tm_hour;
minutes = timeStruct->tm_min;
seconds = timeStruct->tm_sec;
day = timeStruct->tm_mday;
month = timeStruct->tm_mon;
year = timeStruct->tm_year +1900;
```

Imatge 2: *main.c* de l'exemple *RealTimeClock*

3. Gestió de l'entrada d'usuari

En aquest apartat se'ns demana executar cinc exemples, anem a mostrar captures dels mateixos i explicar la seua funcionalitat.

keyboard_async



Imatge 3: Sortida *keyboard_async*

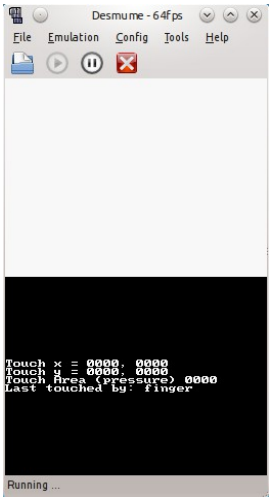
Aquest exemple no és més que un simple teclat sense cap tipus de funcionalitat llevat de poder escriure a la pròpia pantalla de la NDS.

keyboard_stdin

Imatge 4: Sortida *keyboard_stdin*

Al contrari que l'exemple anterior, aquest és un programa que ens pregunta el nostre nom. Una vegada l'introduïm amb el teclat (el mateix que a l'exemple anterior) aquest desapareix i ens contesta *"Hello (nom)"*.

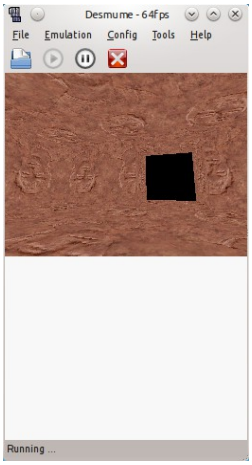
touch_area



Imatge 5: Sortida *touch_area*

Aquest exemple ens diu el punt que hem tocat de la pantalla tàctil, així com la pressió i, depenent de l'àrea, ens diu si hem tocat la pantalla amb el dit o l'estilus (en aquest cas ratolí ja que és un emulador).

touch_look



Imatge 6: Sortida *touch_look*

Ací estem a un entorn 3D pel qual podem rotar la càmera (però no desplaçar-la) tocant la pantalla inferior. Si llisquem cap a l'esquerra, la càmera gira cap a l'esquerra, etc. El moviment és completament lliure.

touch_test

Imatge 7: Sortida *touch_test*

Per acabar, aquest últim exemple ens diu les coordenades on hem tocat la pantalla tàctil i, a més a més, deixa l'sprite d'un punt roig a eixes coordenades. És molt similar a l'exemple *touch_area* llevat del punt roig i que, aquest, no ens diu la pressió ni si hem tocat amb el dit o no.

4. Treball autònom

Com a treball autònom se'ns demana crear un programa on l'usuari puga triar quines dades emmagatzemades en la consola, vol visualitzar.

Adjuntem el codi font i l'arxiu *Makefile* amb aquest exercici resolt. Ací podem veure una captura del programa en execució.

Salvador Llobet, Ignacio
Soria Magraner, Enrique

Imatge 8: Sortida del treball autònom