TEC | Tecnológico de Costa Rica

Área Académica de Ingeniería en Computadores

Curso: CE-5303 Introducción a los sistemas embebidos

Proyecto 2: Herramientas de ingeniería

Profesor: M. Luis Chavarría Zamora

Integrantes:

Calderón Yock Sebastián - 2018161630

Acevedo Rodríguez Kevin - 2018148661

Camacho Hernández Julián - 2019201459

Cartago 2023

Para el desarrollo del proyecto se utilizaron diversas herramientas de ingeniería que facilitaron el desarrollo en general de sus componentes. En este documento se describirán las herramientas más relevantes, agrupadas por el componente para el que fueron empleadas. Los componentes a describir son los mismos identificados en el documento de diseño del proyecto:

Raspberry Pi con Aplicación Bare-Metal

- Raspberry Pi: es una serie de computadoras de placa única (Single Board Computer, SBC). Ofrece un conjunto de pines GPIO que permiten la interacción con componentes electrónicos externos. En el contexto de este proyecto, se utiliza una Raspberry Pi 2 como el núcleo del sistema, ejecutando una aplicación bare-metal en C++.

- Circle: el proyecto Circle proporciona un entorno de nivel de metal C++ para las computadoras de placa única (SBC) Raspberry Pi. Se trata de un marco para desarrollar aplicaciones que se ejecutan directamente en el hardware, sin utilizar un sistema operativo, lo que es en cierta medida equivalente a programar un microcontrolador muy potente. [1] En este proyecto fue utilizado para manejar toda la aplicación baremetal, incluyendo la comunicación UART, el despliegue por medio de HDMI y el timer del sistema.

  Circle utiliza el siguiente toolchain de compilación cruzada:

  - arm-none-eabi-: conjunto de herramientas de desarrollo cruzado que se utiliza para compilar y ensamblar código dirigido a microcontroladores y procesadores basados en la arquitectura ARM, sin un sistema operativo subyacente.

```
PREFIX = /path/to/your/toolchain/bin/arm-none-eabi-
AARCH = 32
RASPPI = 3
```

Figura 1. Archivo de configuración para la compilación cruzada.

- Make: herramienta utilizada para automatizar el proceso de compilación de programas los programas de *Circle* a partir de su código fuente. El archivo Makefile, especifica cómo se deben compilar y enlazar los componentes de un proyecto. Make facilitó la construcción de la imagen *kernel7.img*, lo que ahorró tiempo en el proceso de desarrollo.

app > Makefile

```
1  CIRCLEHOME = ../..
2
3  OBJS    = main.o kernel.o
4
5  LIBS    = $(CIRCLEHOME)/lib/libcircle.a
6
7  include $(CIRCLEHOME)/Rules.mk
8
9  -include $(DEPS)
```

Figura 2. Makefile para generar la imagen ejecutable.

- Scripts: para automatizar diversas tareas durante el desarrollo, como generación de archivos, limpieza, renombramiento de variables, entre otros, se utilizaron scripts de extensión .sh que automatizará esta tarea. Esto agilizó considerablemente el proceso de desarrollo. En la siguiente figura se presenta un ejemplo de script desarrollado:

u-boot-rpi2 > deploy.sh

```
1  #!/bin/bash
2
3  while true; do
4      echo "Seleccione una opción:"
5      echo "1. Copiar los archivos para la aplicación bare-metal."
6      echo "2. Copir los archivos para la desmostración de u-boot."
7      echo "S. Salir."
8
9      read -p "Ingrese su elección: " choice
10
11     case $choice in
12         1)
13             # Montar la particion de la SD en una particion local
14             #sudo mount /dev/sda1 /mnt/sdcard
15             sudo mount /dev/mmcblk0 /mnt/sdcard
16             # Eliminar el config.txt actual
17             sudo rm /mnt/sdcard/config.txt
18             # Copiar el config.txt de la carpeta bare-metal
19             sudo cp bare-metal/config.txt /mnt/sdcard
20             # Desmontar la partición de la SD
21             sudo umount /mnt/sdcard
22             ;;
```

Figura 3. Script para cargar archivos a la tarjeta SD.

Sistema con ADC y comunicación UART

- Arduino: plataforma de hardware de código abierto que consiste en una placa de circuito impreso que integra un microcontrolador, una serie de pines de entrada/salida (GPIO), y una interfaz de programación. En este proyecto se utilizó el ADC que tiene integrado para realizar la lectura del sensor de

proximidad.

- Arduino IDE: software de código abierto que proporciona un entorno de programación y desarrollo para la plataforma Arduino. Fue utilizado para desarrollar el código que se encarga realizar la lectura del sensor para medir la distancia en centímetros.

proyecto2 §

```
#define ECHO_PIN A0
#define TRIGGER_PIN 9

#define SECOND 1000
#define SOUND_SPEED 0.0343

void setup() {
    //! Coincidir con la velocidad de la RaspberryPi
    Serial.begin(115200);
    pinMode(TRIGGER_PIN, OUTPUT);
    pinMode(ECHO_PIN, INPUT);
}

void loop() {
    generate_ultrasonic_pulse();
    long distance = measure_distance();
    Serial.println(distance);
    //Serial.write(distance);
    delay(SECOND);
}
```

Figura 4. Código .ino en el Arduino IDE.

Circuito con sensor

- Sensor ultrasónico: sensor que mide la distancia mediante el uso de ondas ultrasónicas. El cabezal emite una onda ultrasónica y recibe la onda reflejada que retorna desde el objeto. Los sensores ultrasónicos miden la distancia al objeto contando el tiempo entre la emisión y la recepción. [2]

Bootloader con u-boot

- U-boot: se desarrolla mediante herramientas de desarrollo cruzado *arm-none-eabi-*. Su función principal es permitir el arranque de la aplicación bare-metal en la Raspberry Pi 2. La elección de U-Boot garantiza una inicialización eficiente y confiable del sistema, ofreciendo flexibilidad y adaptabilidad al entorno específico de la Raspberry Pi.

Referencias

[1] *'Circle'*. (2019, March 9). 'Circle Project' . Retrieved October 10, 2023, from https://circle-rpi.readthedocs.io/en/45.3/introduction.html

[2] Guía de sensores para fábricas clasificados por principios Fundamentos del sensor. (2023) Retrieved October 10, 2023, from https://www.keyence.com.mx/ss/products/sensor/sensorbasics/ultrasonic/info/