# Instituto Tecnológico de Costa Rica

## Área Académica de Ingeniería en Computadores

*(Computer Engineering Academic Area)*

## Programa de Licenciatura en Ingeniería en Computadores

*(Licentiate Degree Program in Computer Engineering)*

**Curso: CE-4302 Arquitectura de Computadores II**

(Course: CE-4302 Computer Architecture II)


**Evaluación Taller 4: OpenMP, parte II**

*(Workshop 4 evaluation: OpenMP, part II)*

**Profesor:**

*(Professor)*

**Ing. M.Sc. Jeferson González Gómez**

**Fecha: 2 de abril de 2019**

*( Date)*

**Evaluación Taller 4. OpenMP, parte II**

# Introducción

En este taller se estudiará la implementación de paralelismo a partir de un código secuencial en C, utilizando la extensión de OpenMP para dispositivos móviles multiprocesador basados en Android.

# Compilación y ejecución en Android

Para compilar el código y ejecutar la aplicación para plataformas Android debe seguir los siguentes pasos.

1. Instalar la herramienta Android NDK que permite implementar código en C para dispositivo Android. Se recomienda descomprimir el archivo y mover la carpeta generada al directorio /opt/:

   ```
   unzip  android-ndk-r16b-linux-x86_64.zip
   sudo mv android-ndk-r16b /opt/
   ```

2. Instalar Android Debug Bridge (ADB), que posteriormente permitirá llevar la aplicación compilada al dispositivo móvil. Para esto, desde Ubuntu, puede realizarse:

   ```
   sudo apt-get install android-tools-adb android-tools-fastboot
   ```

3. Crear una carpeta llamada *jni* dentro de la cuál debe copiar el código fuente de la aplicación diseñada (se recomienda la compilación nativa previa para verificar el comportamiento funcional de la misma).

4. Modificar los archivos Android.mk y Application.mk (disponibles en el TecDigital), para ajustarse a su aplicación de la siguiente manera:

   **En Android.mk:**

   ```
   LOCAL_MODULE := pi_par

   LOCAL_SRC_FILES := pi_par.c
   ```

   **En Application.mk**

   ```
   APP_MODULES      := pi_par cpufeatures
   ```

5. Dentro de *jni* copiar los archivos Android.mk y Application.mk modificados

6. Dentro de la carpeta *jni*, ejecutar el comando de compilación de ndk:

   ```
   /opt/android-ndk-r16b/ndk-build
   ```

Este comando debe generar las carpetas *libs* y *obj*, en el directorio padre (afuera) de la carpeta *jni*

7. Enviar la aplicación generada (*pi_par*, por ejemplo) al dispositivo móvil (NOTA: el teléfono debe tener activada la depuración por USB, en el modo desarrollador). Aún dentro de *jni* ejecutar:

   ```
   adb push ../libs/armeabi-v7a/pi_par /data/local/tmp
   ```

8. Ejecutar la aplicación en el dispositivo móvil desde la terminal que provee ADB (aún en Linux):

   ```
   adb shell /data/local/tmp/pi_par
   ```

# Ejercicios prácticos

1. Realice un programa que aplique la operación SAXPY tanto serial (normal) como paralelo (OpenMP + NEON), para al menos tres tamaños diferentes de vectores. Mida y compare el tiempo de ejecución entre ambos. Para el uso de las intrínsecas de NEON, puede referirse a: `http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0205j/BABGHIFH.html`, `https://gcc.gnu.org/onlinedocs/gcc-4.8.1/gcc/ARM-NEON-Intrinsics.html` y `http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0491c/CIHJBEFE.html` . textbfNota: Android NDK posee soporte para la unidad vectorial de NEON, por lo que no requiere establecer ninguna bandera en compilación, ver archivo Android.mk.

2. Realice una aplicación paralelizable que requiera gran cantidad de procesamiento. Su aplicación podría ser una aproximación a una integral, una serie, un conjunto de operaciones, etc. Compare tiempos de ejecución de corrida serial vs corrida paralela para diferentes tamaños de datos.

# Entregables

- En TecDigital: Archivo de texto (pdf, doc, odf, etc) que contenga el link a los archivos de código fuente o captura de pantalla del código, así como comandos para compilación y resultados de ejecución.