**Análisis de Memoria**

1. **Objetivos del Análisis**

El presente análisis de memoria busca:

* Evaluar el uso de memoria del sistema de alarmas durante su ejecución normal
* Evaluar el uso de memoria del sistema durante la ejecución de funciones en especifico
  + Monitoreo de voltaje de batería
  + Admin Mode
    - Incluyendo las funciones relacionadas con el almacenamiento de información en memoria
  + Ciclo principal de funcionamiento de interfaz principal
* Detectar posibles fugas de memoria o crecimiento inesperado en el uso de recursos
* Obtener una estimación del consumo de memoria. Esta información permitirá determinar en cuales dispositivos embebidos es factible la implementación del sistema y en cuáles no.

1. **Metodología**

Se utilizó la herramienta memory\_profiler de Python en conjunto con mprof para generar un perfil de uso de memoria a lo largo del tiempo.

El procedimiento incluyó los siguientes pasos:

* Instrumentación del código para incluir las siguientes líneas de código en las funciones de interés:

from memory\_profiler import profile

@profile

* Ejecución del Código mediante mprof:

mprof run main.py

mprof plot

* Ejecución del script principal bajo condiciones normales de operación.
* Ejecución del script en casos específicos de operación.
* Visualización del gráfico de uso de memoria para observar el comportamiento dinámico del sistema.
* Análisis del comportamiento observado.

1. **Resultados**

En primer lugar, la ejecución en condiciones normales del sistema de alarmas. Durante esta ejecución, se ingresó el usuario, se entró a diferentes modos de operación, se activaron alarmas debido a Incendio, Pánico y Fuente de Alimentación.

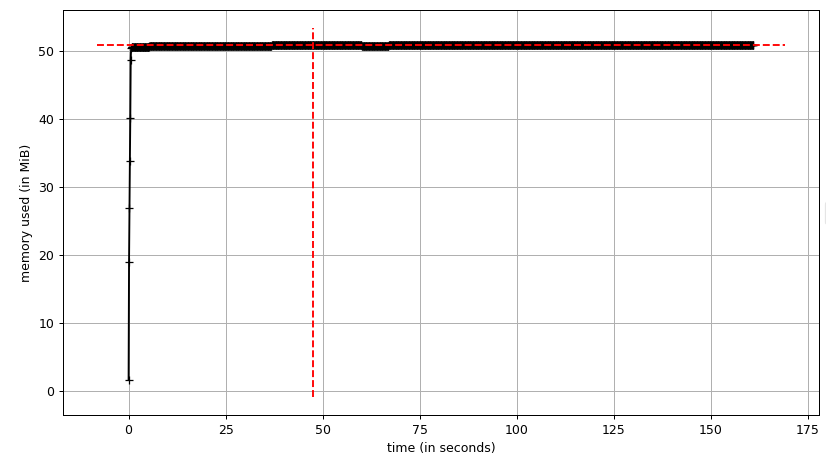


Fig.1 Análisis de uso de memoria en tiempo de ejecución normal

En la Fig. anterior puede encontrarse que luego de un breve periodo de inicialización del sistema, el mismo se estabiliza alrededor de 50 MiB (esto en términos de memoria utilizada) y que no existen saltos inesperados o fugas de memoria detectadas durante el periodo de ejecución.

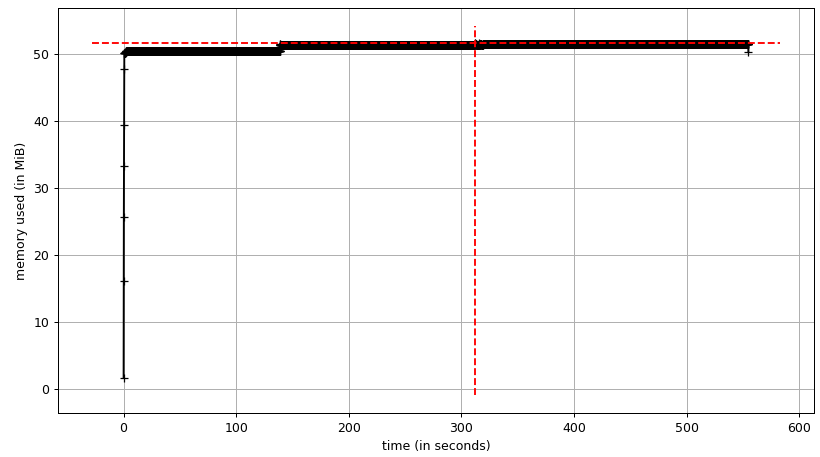


Fig.2 Análisis de uso de memoria en tiempo de ejecución normal durante aproximadamente 8-10 minutos

En la Fig.2 se encuentra también un análisis del uso de memoria durante un lapso mayor, aproximadamente de 10 minutos.

En este caso se observa que la utilización de memoria continua en el mismo rango de valores (50MiB). Sin embargo, a partir del segundo 150 se observa un leve incremento en la utilización. Es recomendable dirigir estudios con lapsos de ejecución mayores para descartar posibles incrementos inesperados.

Seguidamente, el estudio de funciones a cargo de funcionalidades particulares del sistema, como el monitoreo de las fuentes de voltaje, entre otros.

Iniciando con la interfaz de monitoreo de fuentes de voltaje, se obtienen los siguientes resultados.

|  |
| --- |
| Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.3 MiB 52.3 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.3 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.3 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.3 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 label\_bateria.config(bg="green", text="Batería")  91 else:  92 52.3 MiB 0.0 MiB 1 label\_bateria.config(bg="white", text="Batería")  Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.3 MiB 52.3 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.3 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.3 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.3 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 52.3 MiB 0.0 MiB 1 label\_bateria.config(bg="green", text="Batería")  91 else:  92 label\_bateria.config(bg="white", text="Batería")  Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.3 MiB 52.3 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.3 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.3 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.3 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 label\_bateria.config(bg="green", text="Batería")  91 else:  92 52.3 MiB 0.0 MiB 1 label\_bateria.config(bg="white", text="Batería")  Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.3 MiB 52.3 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.3 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.3 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.3 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 52.3 MiB 0.0 MiB 1 label\_bateria.config(bg="green", text="Batería")  91 else:  92 label\_bateria.config(bg="white", text="Batería") |

En los resultados anteriores, se muestran tomas de decisiones diferentes del sistema, cuando el nivel de voltaje de la batería se encuentra sobre el nivel umbral definido y cuando se encuentra por debajo del mismo. Dentro de esto, el sistema se encuentra desarmado y sin alertas diferentes.

Y mediante este estudio de utilización de memoria detallado por función, se observa el comportamiento estable y predecible por parte de la función.

Seguidamente, se activó el Modo Armado del Sistema de Alarmas en Modo 0, para poder añadir un nivel mayor de exigencia a la ejecución. Adicionalmente, se generaron alertas para incrementar el posible uso de memoria.

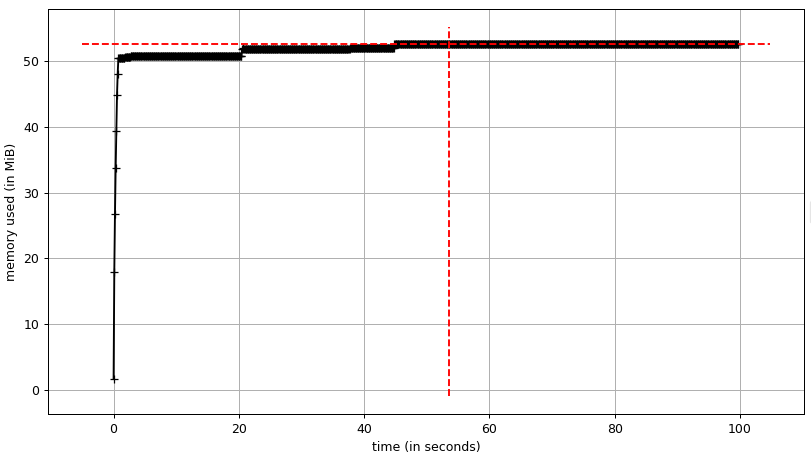


Fig.1 Análisis de uso de memoria en tiempo de ejecución normal con sistema armado en Modo 0.

En la Fig.3 se observa una utilización mayor de memoria en comparativa con las Fig.1 y 2. Sin embargo, el cambio no es desproporcionado y se ajusta a la mayor exigencia establecida para el sistema.

Adicionalmente, en el profiling del sistema se observa que los cambios también son menores:

|  |
| --- |
| Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.5 MiB 52.5 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.5 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.5 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.5 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 52.5 MiB 0.0 MiB 1 label\_bateria.config(bg="green", text="Batería")  91 else:  92 label\_bateria.config(bg="white", text="Batería")  Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.5 MiB 52.5 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.5 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.5 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.5 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 52.5 MiB 0.0 MiB 1 label\_bateria.config(bg="green", text="Batería")  91 else:  92 label\_bateria.config(bg="white", text="Batería")  Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.5 MiB 52.5 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  85 52.5 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.5 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 52.5 MiB 0.0 MiB 1 label\_bateria.config(bg="green", text="Batería")  91 else:  92 label\_bateria.config(bg="white", text="Batería")  Filename: main.py  Line # Mem usage Increment Occurrences Line Contents  =============================================================  80 52.5 MiB 52.5 MiB 1 @profile  81 def verify\_PS(\*args):  82 global current\_PS, ps\_to\_be\_displayed, usr\_Bat\_limit, current\_PS  83 """Función se ejecuta cada vez que el nivel de voltaje cambia"""  84 52.5 MiB 0.0 MiB 1 bat\_lvl\_to\_be\_displayed.set(f"Nivel Bateria: {nivel\_bateria.get()}%")  85 52.5 MiB 0.0 MiB 1 ps\_to\_be\_displayed.set(f"Fuente Alimentación: {current\_PS}")  86  87 # SW-Req: [SW-ID-2]  88 # SW-Req: [SW-ID-18]  89 52.5 MiB 0.0 MiB 1 if (nivel\_bateria.get() < usr\_Bat\_limit and current\_PS == CODE\_BATERIA):  90 52.5 MiB 0.0 MiB 1 label\_bateria.config(bg="green", text="Batería")  91 else:  92 label\_bateria.config(bg="white", text="Batería") |

Se encuentra un ligero aumento en la utilización de memoria, pero justificable dentro del marco de la prueba establecida.

1. **Conclusiones**

* El sistema presenta un comportamiento estable y predecible en cuanto al uso de memoria (tanto en el caso general como en los escenarios particulares generados).
* No se detectan fugas de memoria ni uso excesivo de recursos durante su ejecución típica, lo cual es un indicador positivo sobre una futura implementación del sistema en un sistema embebido.
* La herramienta memory\_profiler demuestra ser útil para detectar posibles cuellos de botella si se combina con el decorador @profile para análisis detallado.

1. **Recomendaciones**

* Se recomienda realizar pruebas extendidas (10-30min o más) e incluir escenarios con mayor profundidad y exigencia, de modo que se pueda robustecer el análisis.