GDB and DASHBOARD

1 Dashboard memory watch

Dashboard GDB pozwala na dynamiczne obserwowanie określonych obszarów pamięci. Używając funkcji `dashboard memory watch`, można ustawić obserwację na konkretne adresy pamięci.

```
dashboard memory watch 0x1ff80 128
```

Powyższa komenda skonfiguruje dashboard do obserwacji 128 bajtów pamięci, począwszy od adresu `0x1ff80`.

2 Obserwacja pamięci przy użyciu watch w GDB

Aby obserwować pamięć na niskim poziomie w GDB, można użyć komendy `watch` w połączeniu z rzutowaniem i rozmiarem obserwowanego obszaru. Przykładowo, aby obserwować 128 bajty pamięci zaczynając od adresu `0x1ff80`, użyjemy następującej komendy:

```
watch *(char*)0x1ff80 @ 128
```

Ta komenda spowoduje przerwanie wykonania programu, gdy wartości w obserwowanym obszarze pamięci ulegną zmianie. Użycie rzutowania na typ `(char*)` pozwala na interpretację adresu jako wskaźnika na bajty, a operator `@` określa ilość obserwowanych bajtów.

3 Clear Memory

3.1 Basic Usage

To clear 128 bytes of memory starting from 0x1FF80, use the following script:

```
#./gdb/zero.py

import gdb
for i in range(0, 128): # 128 bajt w
    gdb.execute("set *((char*)0x1FF80 + %d) = 0" % i)
```

Run this script from GDB dashboard by:

```
source gdb/zero.py
```

3.2 Option 2: With Parameters

To dynamically set the starting address, number of bytes, and pattern from the command line, modify the script as follows:

```
#./gdb/zero_with_params.py

import gdb
import sys

# Parse arguments from the GDB command
args = gdb.string_to_argv(gdb.parameter("args"))
if len(args) != 3:
    print("Usage: source gdb/zero_with_params.py <start_address> <num_bytes> <pattern>")
else:
    start_address = int(args[0], 16)  # Convert start address from hex to int
    num_bytes = int(args[1], 16)      # Convert number of bytes from hex to int
    pattern = int(args[2], 16)        # Convert pattern from hex to int

    for i in range(num_bytes):
        gdb.execute("set *((char*)%d + %d) = %d" % (start_address, i, pattern))
```

To run this script with parameters, set GDB's ‘args‘ parameter before sourcing the script:

```
set args 0x1FF80 0x80 0x30
source gdb/zero_with_params.py
```

This allows specifying the starting address, number of bytes to clear, and the pattern to use directly from the GDB command line.