The MAME story:

FROM C TO MODERN C++

MIODRAG MILANOVIĆ

Who am I ?

Working as software developer from October 2000

Experience in C, C++, C#, Java, …

Software architect in Levi9 – Serbia

From April 2012 coordinator of MAME project

Who we are ?

About 50 active developers

Over 200 contributors

Team contains :

- experienced developers (gaming and not gaming related)
- emulation enthusiasts

Community:

- Developers of different experience
- Software dumpers
- Documentation acquirers
- Testers

What is MAME ?

Multiple Arcade Machine Emulator

Nicola Salmoria started project 1997

MESS as sister project

Preservation of software

Accuracy over performance

Misconception about MAME

Made for playing free games

Is game itself

Way to sell new arcade cabinets

Perfect solution for all emulation

Platform for enhancing games

Why preservation of software is important ?

Companies do not backup all their software

Things get lost

Storage mediums are unreliable

It is always easy to find just “good” and “nice” software

It is not possible to buy some software for bit older platforms

What we wish MAME is used for

Learning about old hardware

Understanding concepts from the past

Preserve hardware and software

Developing new software for old hardware

Relive your childhood

Practice ground for new C++ features

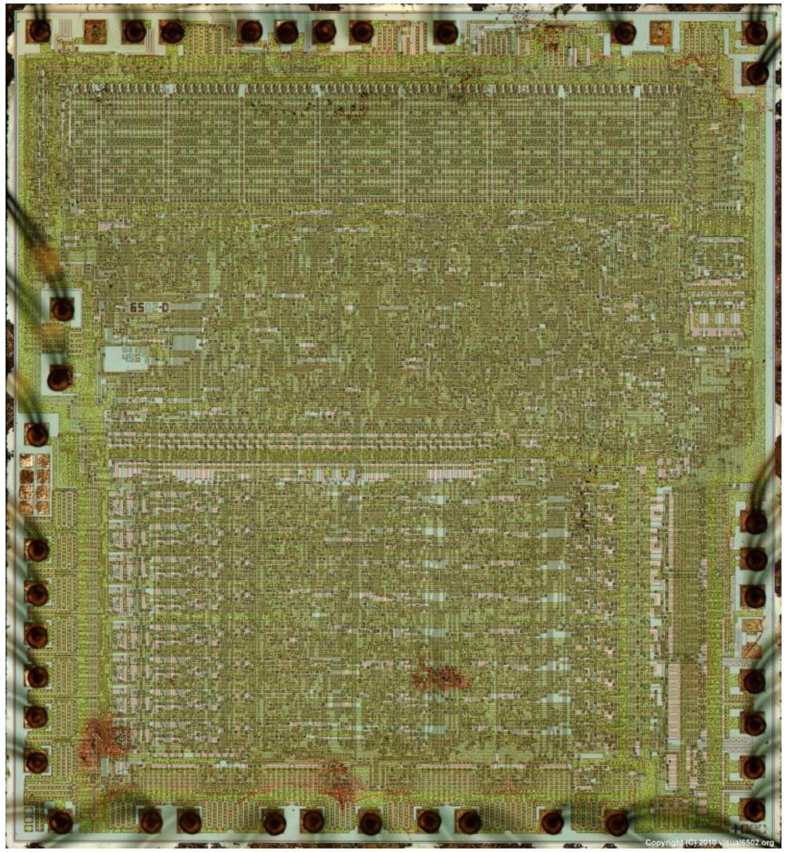
What we do not do ?

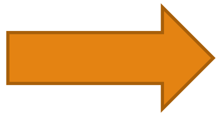
Do not emulate recent hardware (if not permitted by author(s))

Do not support recent software unless permitted (min 3 year after end of production)

Do not try to improve how things look and work

What we actually do ?



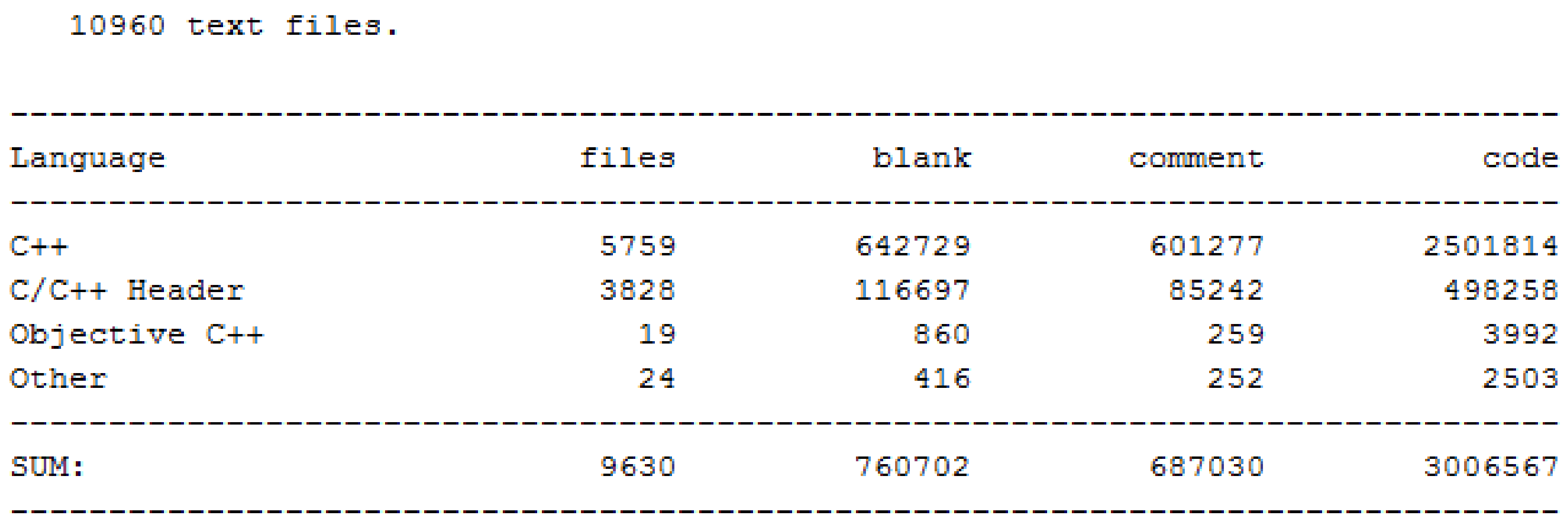


C++ CODE

http://siliconpr0n.org/

http://www.visual6502.org/

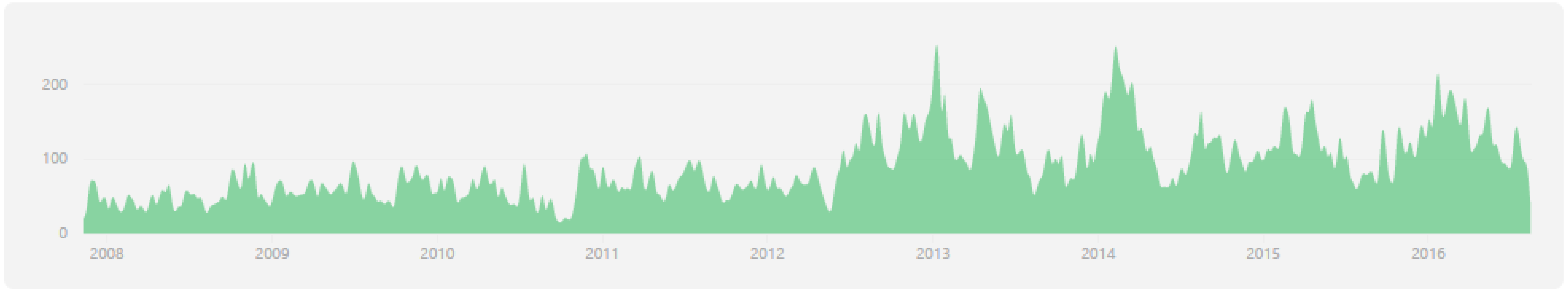
Current statistics



```
   10960 text files.

-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C++                           5759         642729         601277        2501814
C/C++ Header                  3828         116697          85242         498258
Objective C++                   19            860            259           3992
Other                           24            416            252           2503
-------------------------------------------------------------------------------
SUM:                          9630         760702         687030        3006567
-------------------------------------------------------------------------------
```

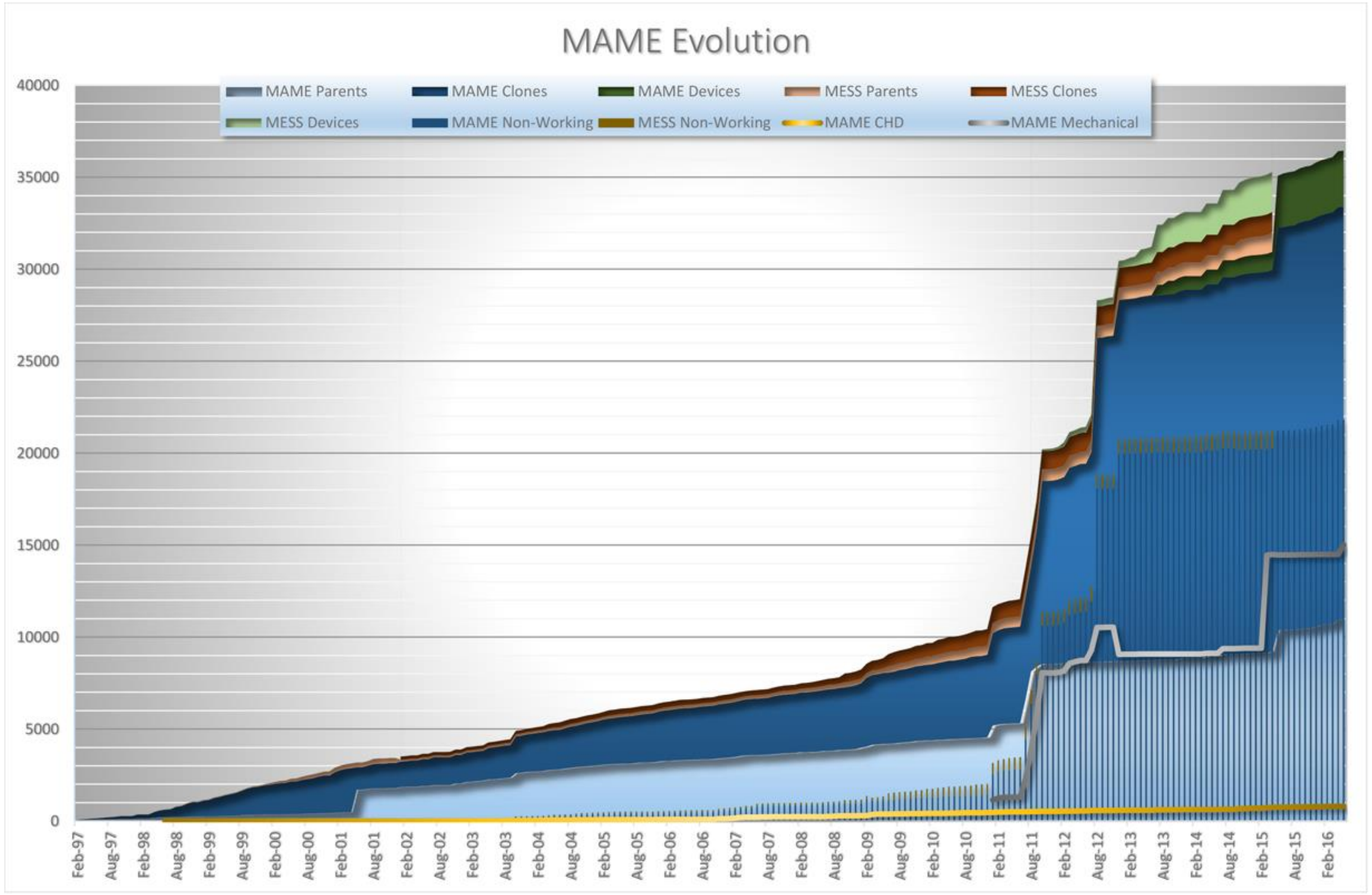
C++ source size : 151 M

GitHub trending



46,610 commits 13 branches 211 releases 165 contributors

Growth trending



Development Tools

On Windows:

- MSYS2 + MinGW-w64 (note that all other distributions do not have threading support)
- MSYS2 + Clang
- Visual Studio 2015

On Linux/FreeBSD:

- GCC 5.x +
- Clang 3.5 +

On OSX / macOS :

- Xcode
- Clang

Commercial tools

ReSharper C++ + Visual Studio 2015 = Ultimate Development Windows Tool

- Do code analysis part by part
- Document your code where applicable

PVS Studio

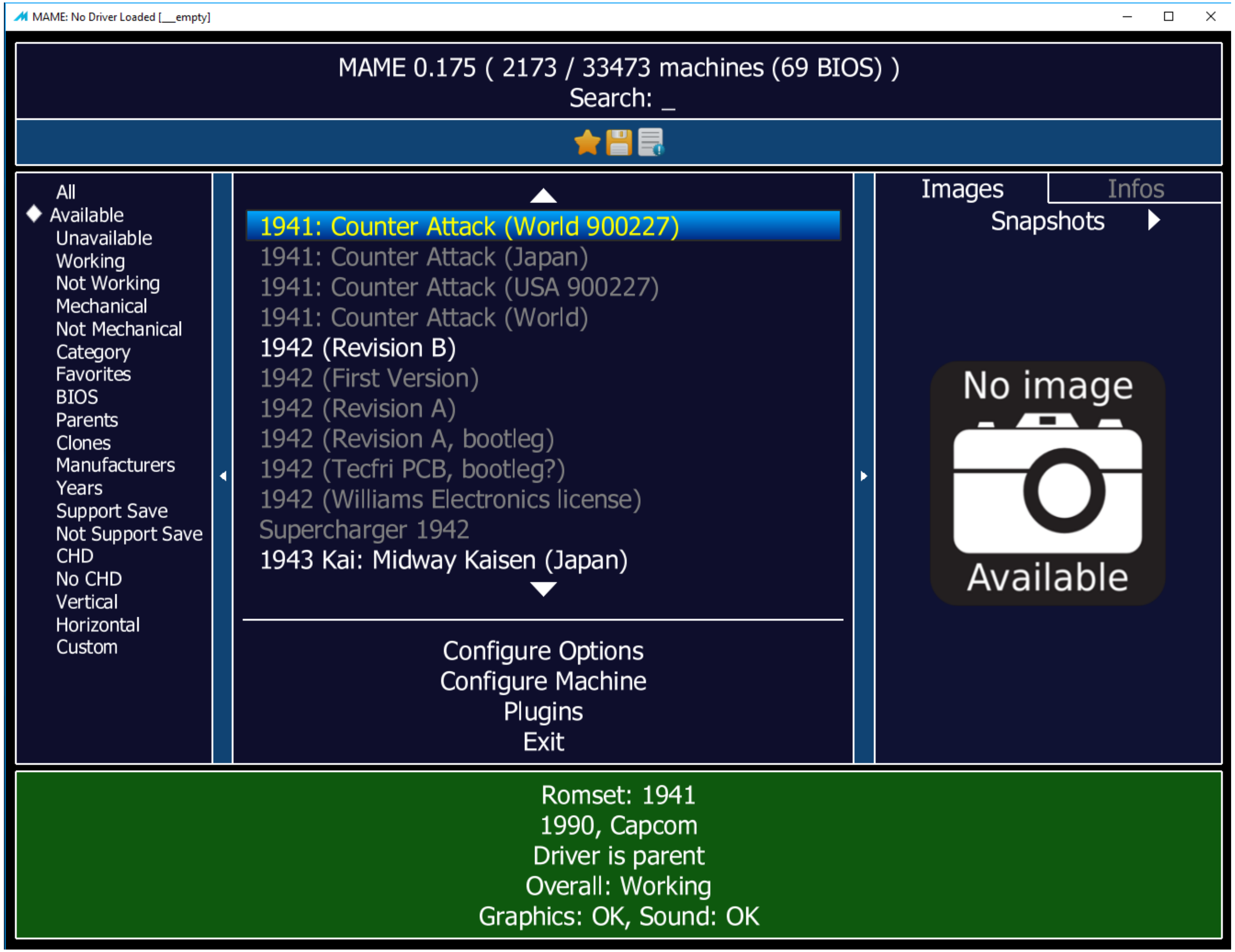
- Static Code Analyzer for C, C++ and C#
- Used evaluation version
- Have been used as one of test cases
- **Software diseases: memset** (http://www.viva64.com/en/b/0154/ from 2012)

Why do we do it ?

"We do these things not because they are easy, but because they are hard“

-John F. Kennedy, 1962

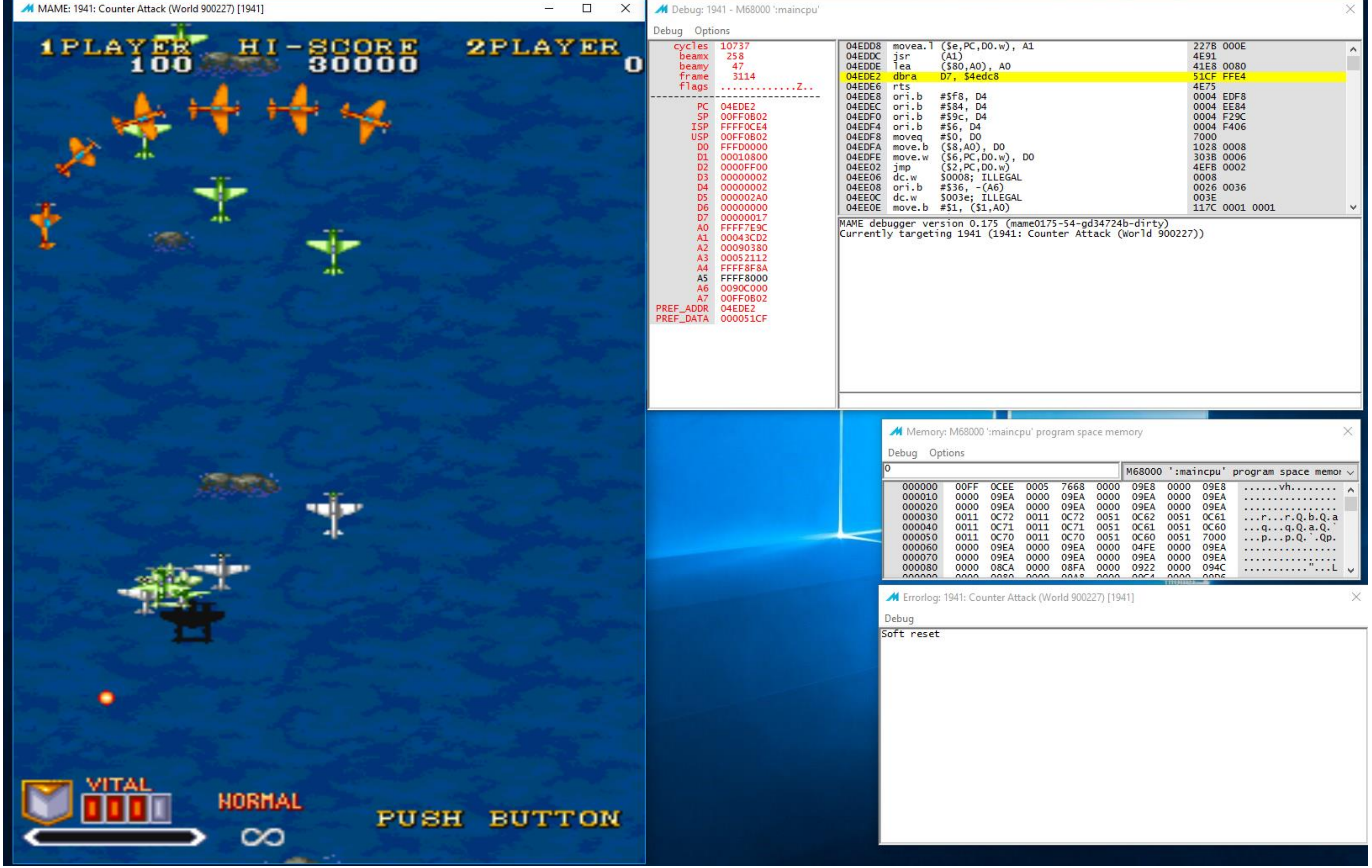
Let’s see how MAME looks


MAME: No Driver Loaded [__empty]

MAME 0.175 (2173 / 33473 machines (69 BIOS))
Search: _

1941: Counter Attack (World 900227)
1941: Counter Attack (Japan)
1941: Counter Attack (USA 900227)
1941: Counter Attack (World)
1942 (Revision B)
1942 (First Version)
1942 (Revision A)
1942 (Revision A, bootleg)
1942 (Tecfri PCB, bootleg?)
1942 (Williams Electronics license)
Supercharger 1942
1943 Kai: Midway Kaisen (Japan)

Romset: 1941
1990, Capcom
Driver is parent
Overall: Working
Graphics: OK, Sound: OK


MAME: 1941: Counter Attack (World 900227) [1941]
1PLAYER
100
HI-SCORE
30000
2PLAYER
0
VITAL
NORMAL
∞
PUSH BUTTON
Debug: 1941 - M68000 ':maincpu'
Debug
Options
Memory: M68000 ':maincpu' program space memory
Debug
Options
0
M68000 ':maincpu' program space memor
Errorlog: 1941: Counter Attack (World 900227) [1941]
Debug

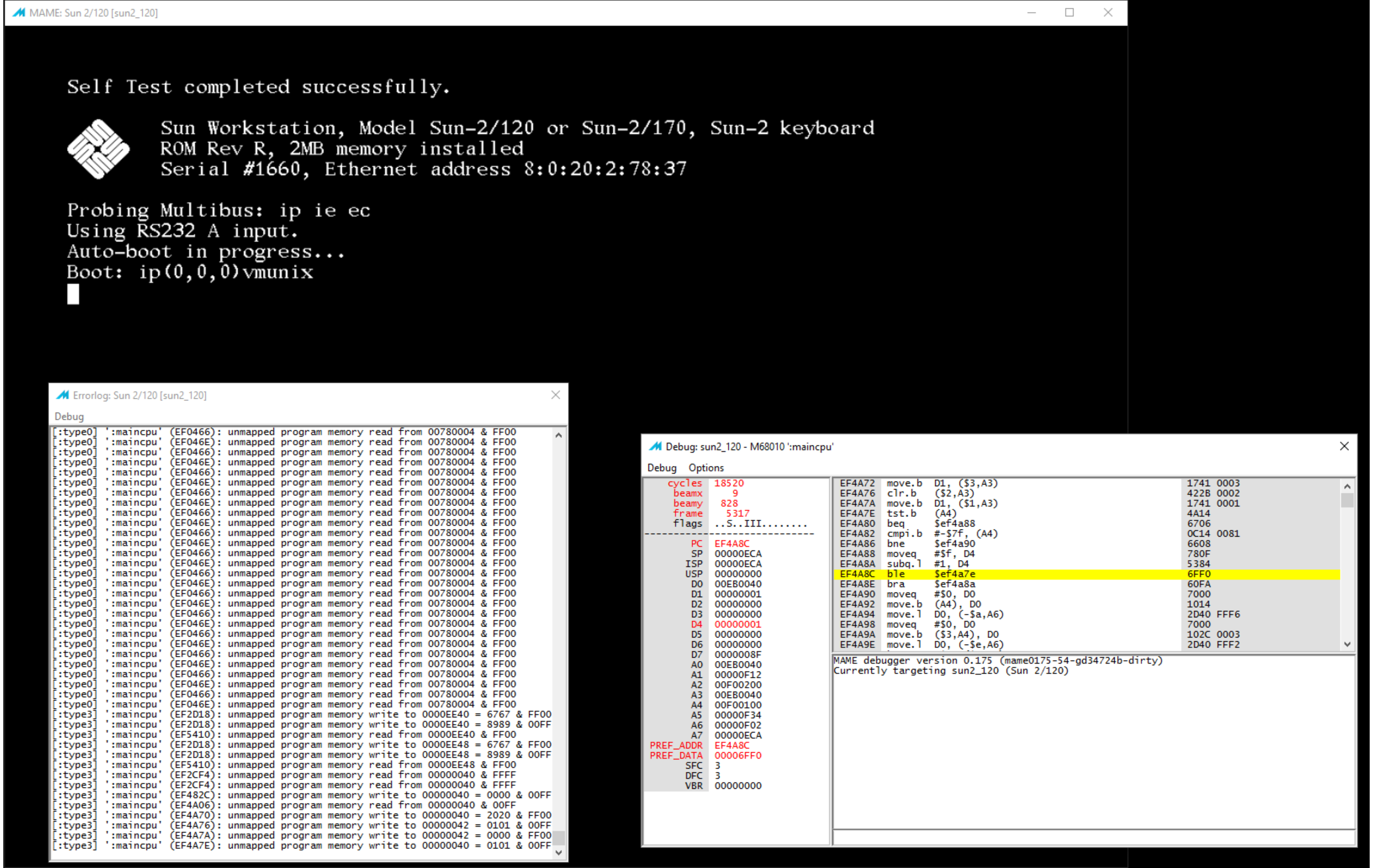
```
    cycles  10737
     beamx   258
     beamy    47
     frame    3114
     flags  .............Z..
-------------------------------
        PC  04EDE2
        SP  00FF0B02
       ISP  FFFF0CE4
       USP  00FF0B02
        D0  FFFD0000
        D1  00010800
        D2  0000FF00
        D3  00000002
        D4  00000002
        D5  000002A0
        D6  00000000
        D7  00000017
        A0  FFFF7E9C
        A1  00043CD2
        A2  00090380
        A3  00052112
        A4  FFFF8F8A
        A5  FFFF8000
        A6  0090C000
        A7  00FF0B02
 PREF_ADDR  04EDE2
 PREF_DATA  000051CF
```

```
04EDD8  movea.l  ($e,PC,D0.w), A1      227B 000E
04EDDC  jsr      (A1)                  4E91
04EDDE  lea      ($80,A0), A0          41E8 0080
04EDE2  dbra     D7, $4edc8            51CF FFE4
04EDE6  rts                            4E75
04EDE8  ori.b    #$f8, D4              0004 EDF8
04EDEC  ori.b    #$84, D4              0004 EE84
04EDF0  ori.b    #$9c, D4              0004 F29C
04EDF4  ori.b    #$6, D4               0004 F406
04EDF8  moveq    #$0, D0               7000
04EDFA  move.b   ($8,A0), D0           1028 0008
04EDFE  move.w   ($6,PC,D0.w), D0      303B 0006
04EE02  jmp      ($2,PC,D0.w)          4EFB 0002
04EE06  dc.w     $0008; ILLEGAL        0008
04EE08  ori.b    #$36, -(A6)           0026 0036
04EE0C  dc.w     $003e; ILLEGAL        003E
04EE0E  move.b   #$1, ($1,A0)          117C 0001 0001
```

```
MAME debugger version 0.175 (mame0175-54-gd34724b-dirty)
Currently targeting 1941 (1941: Counter Attack (World 900227))
```

```
000000  00FF 0CEE 0005 7668 0000 09E8 0000 09E8  ......vh........
000010  0000 09EA 0000 09EA 0000 09EA 0000 09EA  ................
000020  0000 09EA 0000 09EA 0000 09EA 0000 09EA  ................
000030  0011 0C72 0011 0C72 0051 0C62 0051 0C61  ...r...r.Q.b.Q.a
000040  0011 0C71 0011 0C71 0051 0C61 0051 0C60  ...q...q.Q.a.Q.`
000050  0011 0C70 0011 0C70 0051 0C60 0051 7000  ...p...p.Q.`.Qp.
000060  0000 09EA 0000 09EA 0000 04FE 0000 09EA  ................
000070  0000 09EA 0000 09EA 0000 09EA 0000 09EA  ................
000080  0000 08CA 0000 08FA 0000 0922 0000 094C  ..........."...L
```

```
Soft reset
```



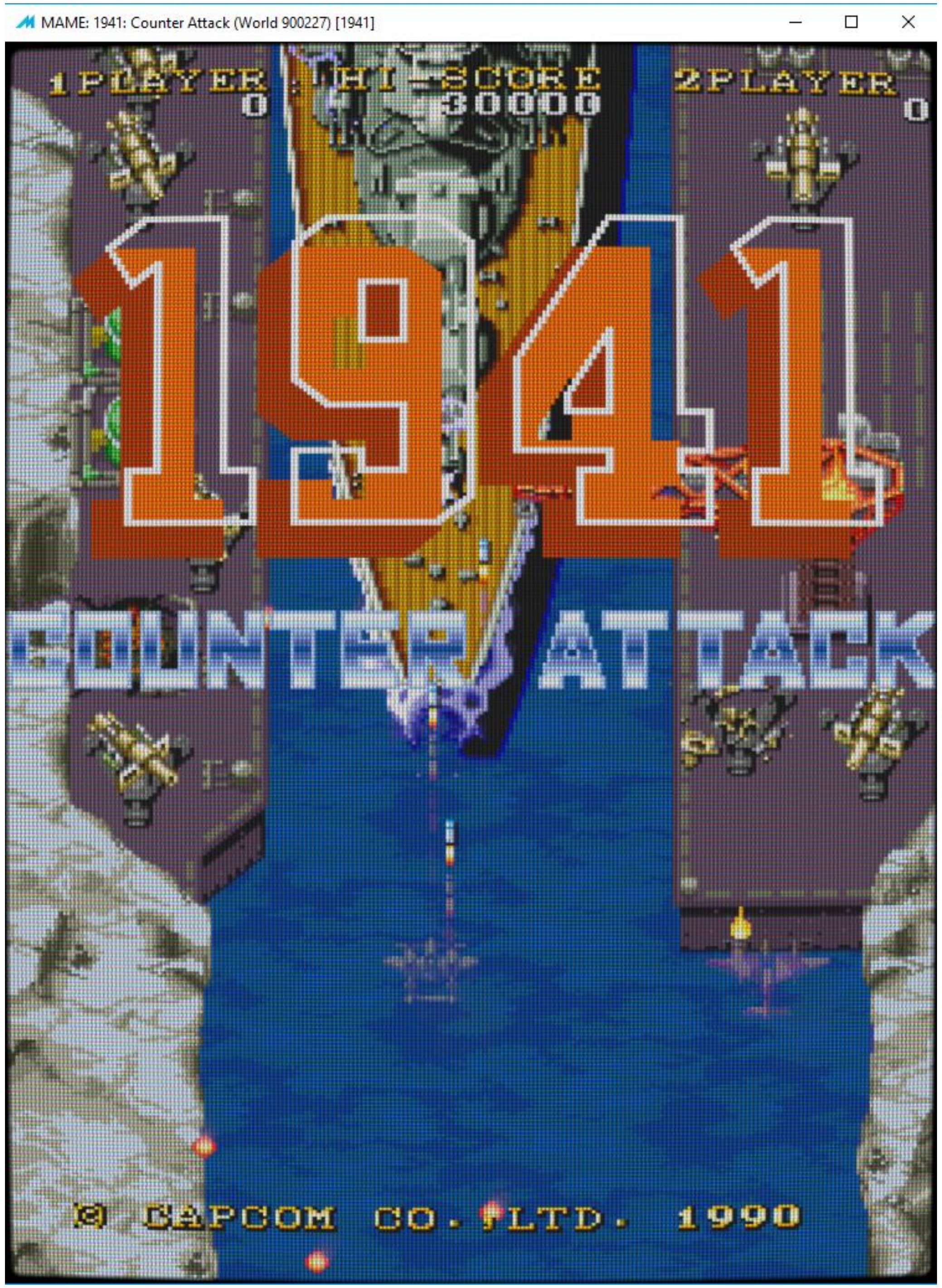
```
Self Test completed successfully.

     Sun Workstation, Model Sun-2/120 or Sun-2/170, Sun-2 keyboard
     ROM Rev R, 2MB memory installed
     Serial #1660, Ethernet address 8:0:20:2:78:37

Probing Multibus: ip ie ec
Using RS232 A input.
Auto-boot in progress...
Boot: ip(0,0,0)vmunix
█
```

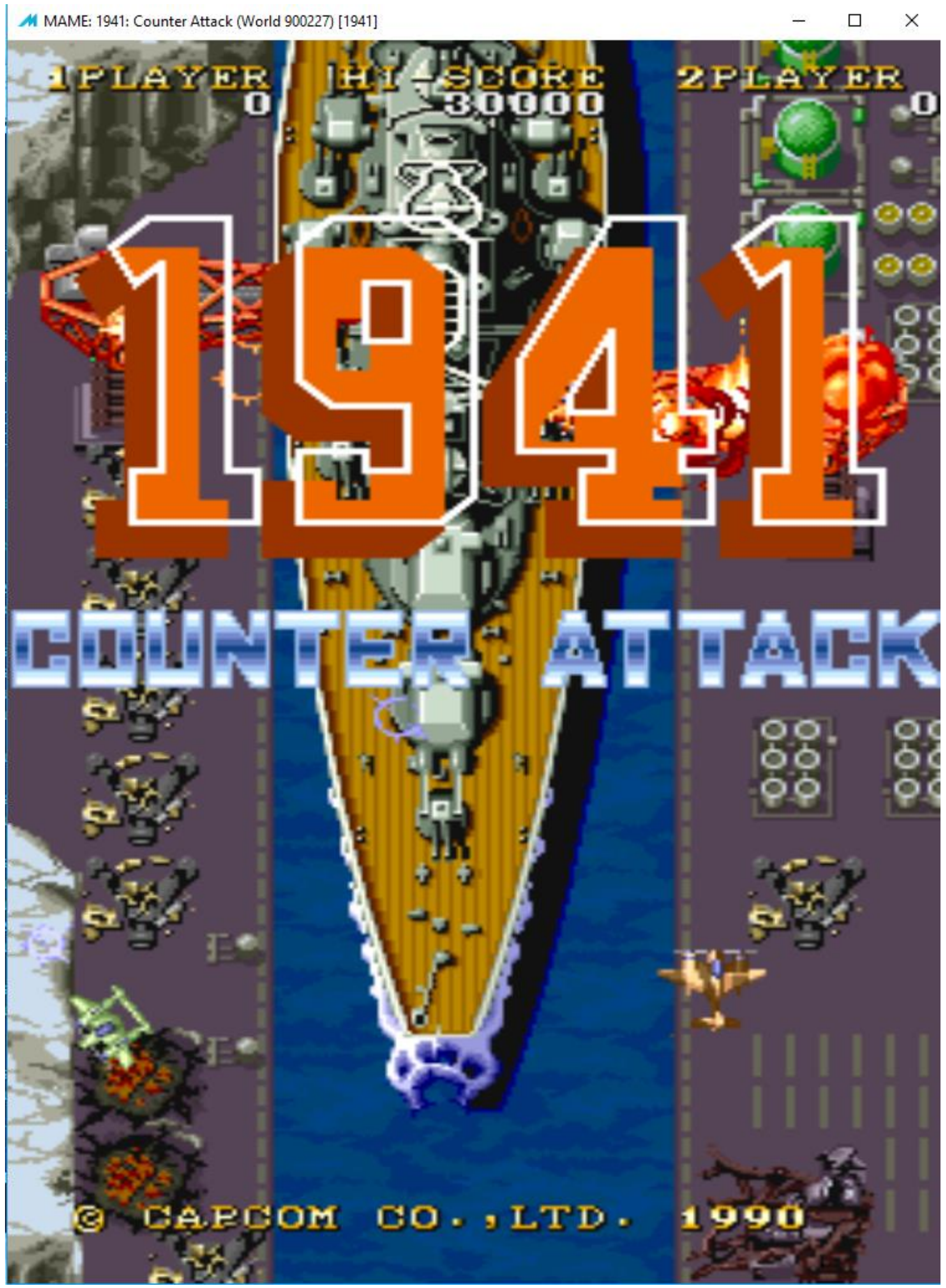
```
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF0466): unmapped program memory read from 00780004 & FF00
[:type0] ':maincpu' (EF046E): unmapped program memory read from 00780004 & FF00
[:type3] ':maincpu' (EF2D18): unmapped program memory write to 0000EE40 = 6767 & FF00
[:type3] ':maincpu' (EF2D18): unmapped program memory write to 0000EE40 = 8989 & 00FF
[:type3] ':maincpu' (EF5410): unmapped program memory read from 0000EE40 & FF00
[:type3] ':maincpu' (EF2D18): unmapped program memory write to 0000EE48 = 6767 & FF00
[:type3] ':maincpu' (EF2D18): unmapped program memory write to 0000EE48 = 8989 & 00FF
[:type3] ':maincpu' (EF5410): unmapped program memory read from 0000EE48 & FF00
[:type3] ':maincpu' (EF2CF4): unmapped program memory read from 00000040 & FFFF
[:type3] ':maincpu' (EF2CF4): unmapped program memory read from 00000040 & FFFF
[:type3] ':maincpu' (EF482C): unmapped program memory write to 00000040 = 0000 & 00FF
[:type3] ':maincpu' (EF4A06): unmapped program memory read from 00000040 & 00FF
[:type3] ':maincpu' (EF4A70): unmapped program memory write to 00000040 = 2020 & FF00
[:type3] ':maincpu' (EF4A76): unmapped program memory write to 00000042 = 0101 & 00FF
[:type3] ':maincpu' (EF4A7A): unmapped program memory write to 00000042 = 0000 & FF00
[:type3] ':maincpu' (EF4A7E): unmapped program memory write to 00000040 = 0101 & 00FF
```

```
     cycles  18520
      beamx      9
      beamy    828
      frame   5317
      flags  ..S..III........
-----------------------------
         PC  EF4A8C
         SP  00000ECA
        ISP  00000ECA
        USP  00000000
         D0  00EB0040
         D1  00000001
         D2  00000000
         D3  00000000
         D4  00000001
         D5  00000000
         D6  00000000
         D7  0000008F
         A0  00EB0040
         A1  00000F12
         A2  00F00200
         A3  00EB0040
         A4  00F00100
         A5  00000F34
         A6  00000F02
         A7  00000ECA
  PREF_ADDR  EF4A8C
  PREF_DATA  00006FF0
        SFC  3
        DFC  3
        VBR  00000000
```

```
EF4A72  move.b  D1, ($3,A3)        1741 0003
EF4A76  clr.b   ($2,A3)            422B 0002
EF4A7A  move.b  D1, ($1,A3)        1741 0001
EF4A7E  tst.b   (A4)               4A14
EF4A80  beq     $ef4a88            6706
EF4A82  cmpi.b  #-$7f, (A4)        0C14 0081
EF4A86  bne     $ef4a90            6608
EF4A88  moveq   #$f, D4            780F
EF4A8A  subq.l  #1, D4             5384
EF4A8C  ble     $ef4a7e            6FF0
EF4A8E  bra     $ef4a8a            60FA
EF4A90  moveq   #$0, D0            7000
EF4A92  move.b  (A4), D0           1014
EF4A94  move.l  D0, (-$a,A6)       2D40 FFF6
EF4A98  moveq   #$0, D0            7000
EF4A9A  move.b  ($3,A4), D0        102C 0003
EF4A9E  move.l  D0, (-$e,A6)       2D40 FFF2
```

```
MAME debugger version 0.175 (mame0175-54-gd34724b-dirty)
Currently targeting sun2_120 (Sun 2/120)
```


MAME: 1941: Counter Attack (World 900227) [1941]
1PLAYER
0
HI-SCORE
30000
2PLAYER
0
1941
COUNTER ATTACK
© CAPCOM CO.,LTD. 1990


MAME: 1941: Counter Attack (World 900227) [1941]
1PLAYER
0
HI-SCORE
30000
2PLAYER
0
1941
COUNTER ATTACK
© CAPCOM CO.,LTD. 1990

How modernization started ?

Plain C project till February 2009

Aaron Giles started conversion to C++

2015 going modern C++

Why converting to C++ ?

Was quite hard to understand code even for existing developers.

Learning curve was bad, so we could not attract new developers.

Adding new functionality was hard, since it side effects were not clear.

Lot of global variables making reuse of specific parts of code impossible.

Code reuse was not clear.

Global symbol pollution was high.

We wish to have “code as documentation” approach.

First steps

Compile your C code as C++

Treat warnings as errors

Use multiple compilers on multiple platforms

OOP

Recognize classes and objects in your code

Recognize connections between them

Object oriented is great and natural way of documenting

Do not create over-engineered model of classes

Express your thoughts

First problems

Global variables

Large number of macros

No tools to help this process

Enforcing coding conventions

Manual labor

Team effort, but keep group working on conversion close

Remove all deprecated code once you remove their usage

Doing one small change you will end up redoing large portion of code

Keep track of changes

Clean/reformat your code

Automatization

Try using REGEX when applicable

Custom made tools – mostly for recognizing pattern of usage and doing replaces

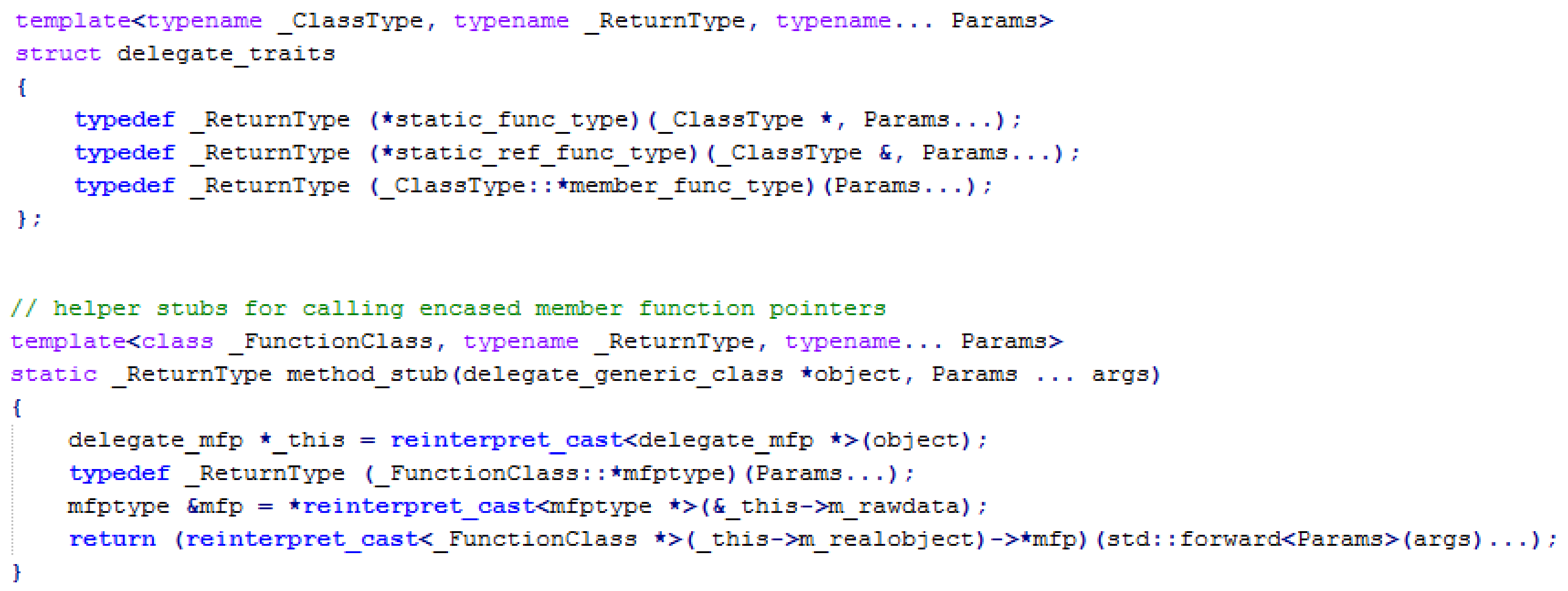
Clang-tidy for modernization (for moving to modern C++)

- **modernize-use-nullptr**
- **modernize-use-override**
- **modernize-use-using**
- **modernize-use-default**
- **modernize-use-bool-literals**
- **modernize-use-auto**
- **modernize-make-unique**

make_unique_clear

```cpp
template<typename Tp> struct MakeUniqClearT { typedef std::unique_ptr<Tp> single_object; };

template<typename Tp> struct MakeUniqClearT<Tp[]> { typedef std::unique_ptr<Tp[]> array; };

template<typename Tp, size_t Bound> struct MakeUniqClearT<Tp[Bound]> { struct invalid_type { }; };

/// make_unique_clear for single objects
template<typename Tp, typename... Params>
inline typename MakeUniqClearT<Tp>::single_object make_unique_clear(Params&&... args)
{
    void *const ptr = ::operator new(sizeof(Tp)); // allocate memory
    std::memset(ptr, 0, sizeof(Tp));
    return std::unique_ptr<Tp>(new(ptr) Tp(std::forward<Params>(args)...));
}

/// make_unique_clear for arrays of unknown bound
template<typename Tp>
inline typename MakeUniqClearT<Tp>::array make_unique_clear(size_t num)
{
    auto size = sizeof(std::remove_extent_t<Tp>) * num;
    unsigned char* ptr = new unsigned char[size]; // allocate memory
    std::memset(ptr, 0, size);
    return std::unique_ptr<Tp>(new(ptr) std::remove_extent_t<Tp>[num]());
}

template<typename Tp, unsigned char F>
inline typename MakeUniqClearT<Tp>::array make_unique_clear(size_t num)
{
    auto size = sizeof(std::remove_extent_t<Tp>) * num;
    unsigned char* ptr = new unsigned char[size]; // allocate memory
    std::memset(ptr, F, size);
    return std::unique_ptr<Tp>(new(ptr) std::remove_extent_t<Tp>[num]());
}

/// Disable make_unique_clear for arrays of known bound
template<typename Tp, typename... Params>
inline typename MakeUniqClearT<Tp>::invalid_type make_unique_clear(Params&&...) = delete;
```

Variadic templates



```
template<typename _ClassType, typename _ReturnType, typename... Params>
struct delegate_traits
{
    typedef _ReturnType (*static_func_type)(_ClassType *, Params...);
    typedef _ReturnType (*static_ref_func_type)(_ClassType &, Params...);
    typedef _ReturnType (_ClassType::*member_func_type)(Params...);
};

// helper stubs for calling encased member function pointers
template<class _FunctionClass, typename _ReturnType, typename... Params>
static _ReturnType method_stub(delegate_generic_class *object, Params ... args)
{
    delegate_mfp *_this = reinterpret_cast<delegate_mfp *>(object);
    typedef _ReturnType (_FunctionClass::*mfptype)(Params...);
    mfptype &mfp = *reinterpret_cast<mfptype *>(&_this->m_rawdata);
    return (reinterpret_cast<_FunctionClass *>(_this->m_realobject)->*mfp)(std::forward<Params>(args)...);
}
```

Constexpr

```
/* Concatenate/extract 32-bit halves of 64-bit values */
constexpr UINT64 concat_64(UINT32 hi, UINT32 lo) { return (UINT64(hi) << 32) | UINT32(lo); }
constexpr UINT32 extract_64hi(UINT64 val) { return UINT32(val >> 32); }
constexpr UINT32 extract_64lo(UINT64 val) { return UINT32(val); }
```

```
#ifdef LSB_FIRST
constexpr UINT16 big_endianize_int16(UINT16 x) { return flipendian_int16(x); }
constexpr UINT32 big_endianize_int32(UINT32 x) { return flipendian_int32(x); }
constexpr UINT64 big_endianize_int64(UINT64 x) { return flipendian_int64(x); }
constexpr UINT16 little_endianize_int16(UINT16 x) { return x; }
constexpr UINT32 little_endianize_int32(UINT32 x) { return x; }
constexpr UINT64 little_endianize_int64(UINT64 x) { return x; }
#else
constexpr UINT16 big_endianize_int16(UINT16 x) { return x; }
constexpr UINT32 big_endianize_int32(UINT32 x) { return x; }
constexpr UINT64 big_endianize_int64(UINT64 x) { return x; }
constexpr UINT16 little_endianize_int16(UINT16 x) { return flipendian_int16(x); }
constexpr UINT32 little_endianize_int32(UINT32 x) { return flipendian_int32(x); }
constexpr UINT64 little_endianize_int64(UINT64 x) { return flipendian_int64(x); }
#endif /* LSB_FIRST */
```

These were macros

And more constexpr

```cpp
// Highly useful template for compile-time knowledge of an array size
template <typename T, size_t N> constexpr size_t ARRAY_LENGTH(T (&)[N]) { return N;}




constexpr UINT16 flipendian_int16(UINT16 val) { return (val << 8) | (val >> 8); }

constexpr UINT32 flipendian_int32_partial16(UINT32 val) { return ((val << 8) & 0xFF00FF00U) | ((val >> 8) & 0x00FF00FFU); }
constexpr UINT32 flipendian_int32(UINT32 val) { return (flipendian_int32_partial16(val) << 16) | (flipendian_int32_partial16(val) >> 16); }

constexpr UINT64 flipendian_int64_partial16(UINT64 val) { return ((val << 8) & U64(0xFF00FF00FF00FF00)) | ((val >> 8) & U64(0x00FF00FF00FF00
constexpr UINT64 flipendian_int64_partial32(UINT64 val) { return ((flipendian_int64_partial16(val) << 16) & U64(0xFFFF0000FFFF0000)) | ((fli
constexpr UINT64 flipendian_int64(UINT64 val) { return (flipendian_int64_partial32(val) << 32) | (flipendian_int64_partial32(val) >> 32); }
```

New features that helped clean

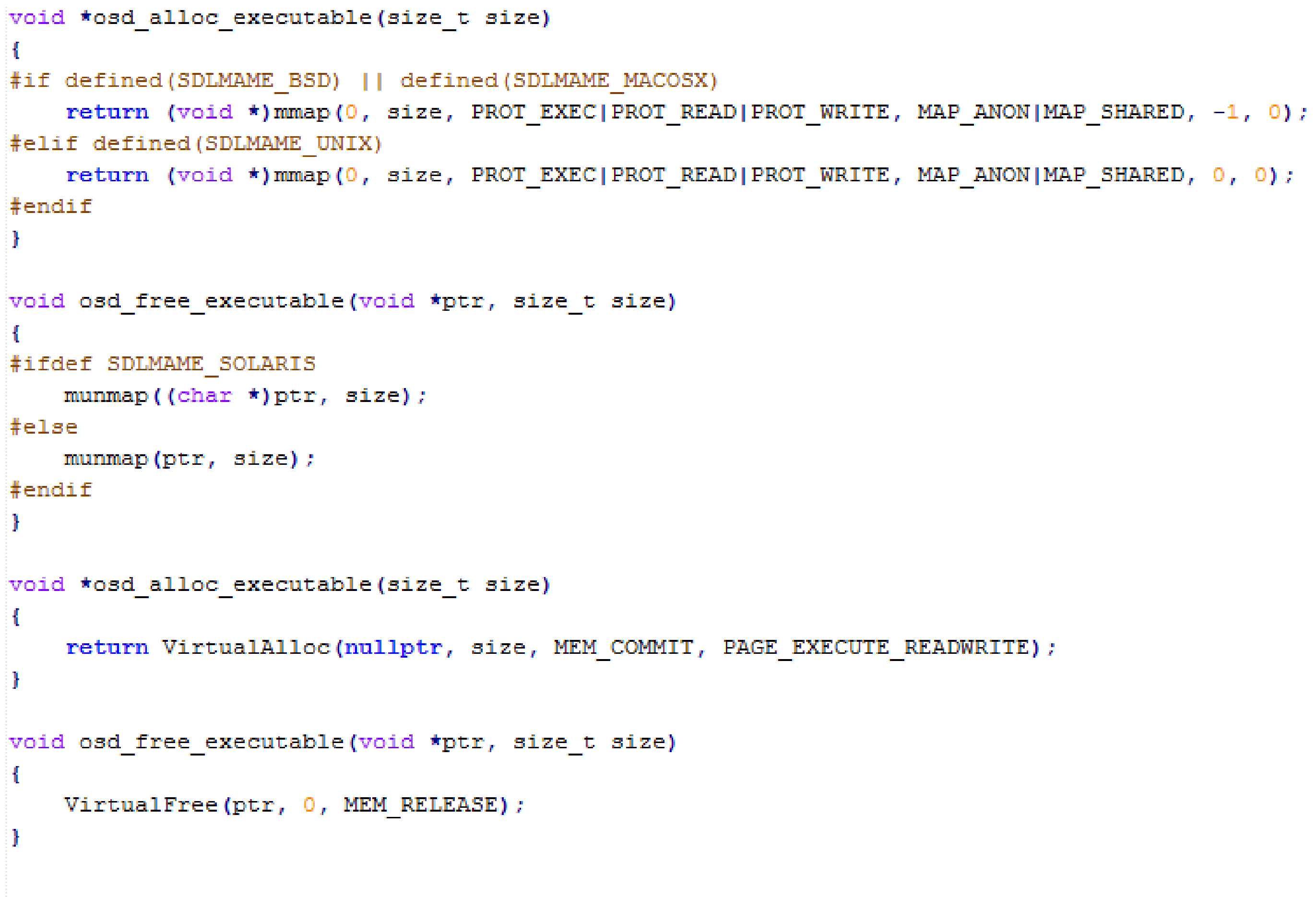
std::mutex

std::thread

atomics

chrono

Still not able to convert



```
void *osd_alloc_executable(size_t size)
{
#if defined(SDLMAME_BSD) || defined(SDLMAME_MACOSX)
    return (void *)mmap(0, size, PROT_EXEC|PROT_READ|PROT_WRITE, MAP_ANON|MAP_SHARED, -1, 0);
#elif defined(SDLMAME_UNIX)
    return (void *)mmap(0, size, PROT_EXEC|PROT_READ|PROT_WRITE, MAP_ANON|MAP_SHARED, 0, 0);
#endif
}

void osd_free_executable(void *ptr, size_t size)
{
#ifdef SDLMAME_SOLARIS
    munmap((char *)ptr, size);
#else
    munmap(ptr, size);
#endif
}

void *osd_alloc_executable(size_t size)
{
    return VirtualAlloc(nullptr, size, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
}

void osd_free_executable(void *ptr, size_t size)
{
    VirtualFree(ptr, 0, MEM_RELEASE);
}
```

Linux / macOS

Windows

Things we wish to use

Coroutines

- Experimented with https://byuu.org/library/libco/ got bad results due to stackful implementation
- Wait state implementation
- For networking layer implemenatation

Modules

- Too big compile times right now (30-40 on latest hardware)
- Ideal since we need quite large amount of definitions (currently using #include “emu.h” in each separate emulator)

GSL

And from the tech side :

Fully working Android build

iOS build

Console build (Xbox One and PS4)

Various VR systems

Hololens

Interaction with real hardware (using IoT devices as proxies)

GENie project generator

https://github.com/bkaradzic/GENie/

GENie (pronounced as Jenny) is project generator tool. It automagically generates project from Lua script, making applying the same settings for multiple projects easy.

Supported project generators:

FASTBuild (experimental)

GNU Makefile

Ninja (experimental)

Qbs / QtCreator (experimental)

Visual Studio 2008, 2010, 2012, 2013, 2015, 15

XCode

Why using GENie ?

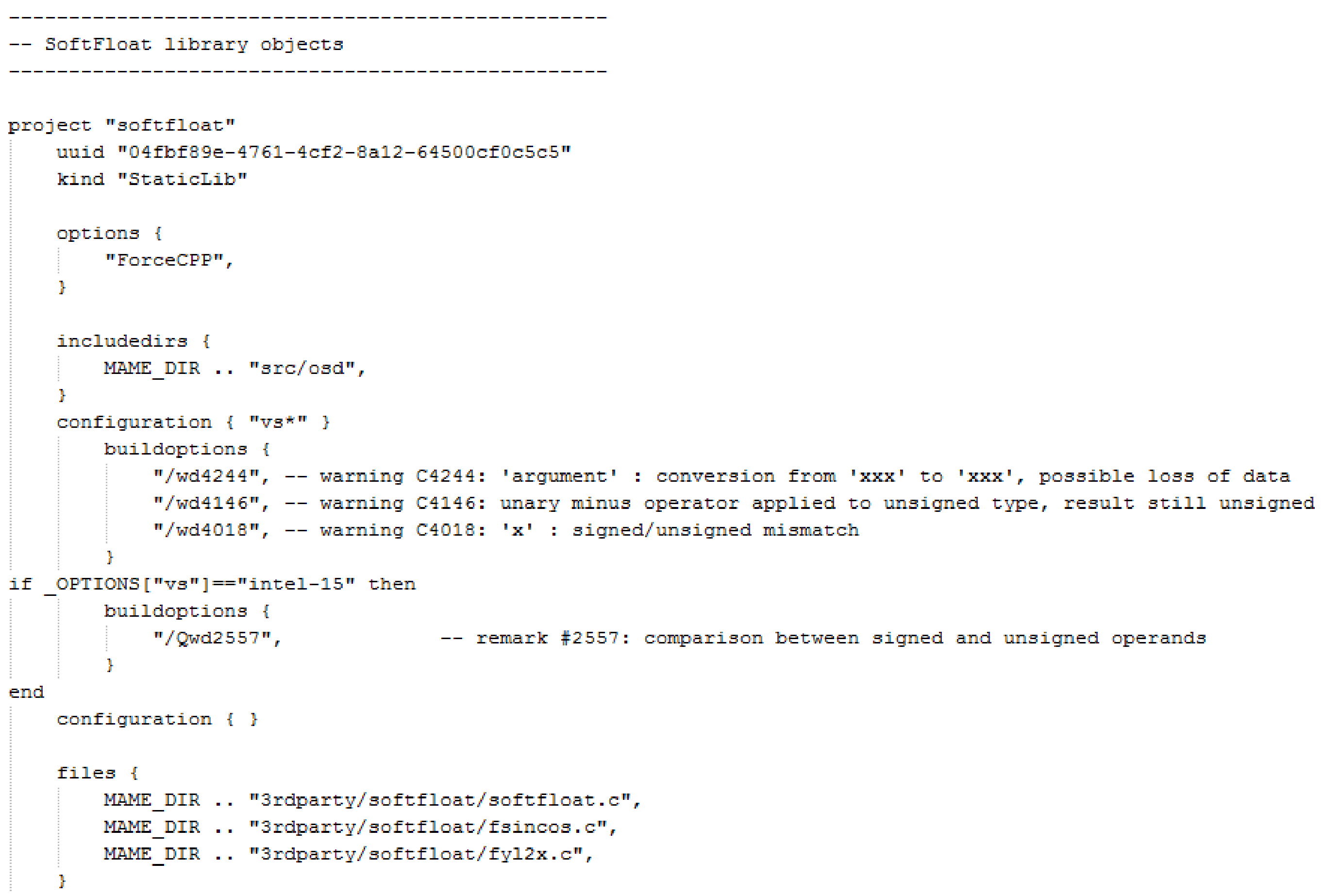
Can generate all compilers we are targeting

Based on LUA

Easy extensible

Enables us to create custom/partial builds

Example usage



```
--------------------------------------------------
-- SoftFloat library objects
--------------------------------------------------

project "softfloat"
    uuid "04fbf89e-4761-4cf2-8a12-64500cf0c5c5"
    kind "StaticLib"

    options {
        "ForceCPP",
    }

    includedirs {
        MAME_DIR .. "src/osd",
    }
    configuration { "vs*" }
        buildoptions {
            "/wd4244", -- warning C4244: 'argument' : conversion from 'xxx' to 'xxx', possible loss of data
            "/wd4146", -- warning C4146: unary minus operator applied to unsigned type, result still unsigned
            "/wd4018", -- warning C4018: 'x' : signed/unsigned mismatch
        }
if _OPTIONS["vs"]=="intel-15" then
        buildoptions {
            "/Qwd2557",             -- remark #2557: comparison between signed and unsigned operands
        }
end
    configuration { }

    files {
        MAME_DIR .. "3rdparty/softfloat/softfloat.c",
        MAME_DIR .. "3rdparty/softfloat/fsincos.c",
        MAME_DIR .. "3rdparty/softfloat/fyl2x.c",
    }
```

How does MAME works?

```
// license:BSD-3-Clause
// copyright-holders:Robbbert

#include "emu.h"
#include "bus/rs232/rs232.h"
#include "cpu/s2650/s2650.h"
#include "machine/terminal.h"
#include "imagedev/snapquik.h"

class pipbug_state : public driver_device
{
public:
    pipbug_state(const machine_config &mconfig, device_type type, const char *tag)
        : driver_device(mconfig, type, tag),
        m_rs232(*this, "rs232"),
        m_maincpu(*this, "maincpu")
    {
    }

    DECLARE_WRITE8_MEMBER(pipbug_ctrl_w);
    required_device<rs232_port_device> m_rs232;
    required_device<cpu_device> m_maincpu;
};

WRITE8_MEMBER( pipbug_state::pipbug_ctrl_w )
{
// 0x80 is written here - not connected in the baby 2650
}

static ADDRESS_MAP_START(pipbug_mem, AS_PROGRAM, 8, pipbug_state)
    ADDRESS_MAP_UNMAP_HIGH
    AM_RANGE( 0x0000, 0x03ff) AM_ROM
    AM_RANGE( 0x0400, 0x7fff) AM_RAM
ADDRESS_MAP_END
```

```
static ADDRESS_MAP_START(pipbug_io, AS_IO, 8, pipbug_state)
    AM_RANGE(S2650_CTRL_PORT, S2650_CTRL_PORT) AM_WRITE(pipbug_ctrl_w)
    AM_RANGE(S2650_SENSE_PORT, S2650_SENSE_PORT) AM_READNOP
ADDRESS_MAP_END

/* Input ports */
static INPUT_PORTS_START( pipbug )
INPUT_PORTS_END

static DEVICE_INPUT_DEFAULTS_START( terminal )
    DEVICE_INPUT_DEFAULTS( "RS232_TXBAUD", 0xff, RS232_BAUD_110 )
    DEVICE_INPUT_DEFAULTS( "RS232_RXBAUD", 0xff, RS232_BAUD_110 )
    DEVICE_INPUT_DEFAULTS( "RS232_STARTBITS", 0xff, RS232_STARTBITS_1 )
    DEVICE_INPUT_DEFAULTS( "RS232_DATABITS", 0xff, RS232_DATABITS_7 )
    DEVICE_INPUT_DEFAULTS( "RS232_PARITY", 0xff, RS232_PARITY_EVEN )
    DEVICE_INPUT_DEFAULTS( "RS232_STOPBITS", 0xff, RS232_STOPBITS_1 )
DEVICE_INPUT_DEFAULTS_END

static MACHINE_CONFIG_START( pipbug, pipbug_state )
    /* basic machine hardware */
    MCFG_CPU_ADD("maincpu",S2650, XTAL_1MHz)
    MCFG_CPU_PROGRAM_MAP(pipbug_mem)
    MCFG_CPU_IO_MAP(pipbug_io)
    MCFG_S2650_FLAG_HANDLER(DEVWRITELINE("rs232", rs232_port_device, write_txd))

    /* video hardware */
    MCFG_RS232_PORT_ADD("rs232", default_rs232_devices, "terminal")
    MCFG_RS232_RXD_HANDLER(INPUTLINE("maincpu", S2650_SENSE_LINE))
    MCFG_DEVICE_CARD_DEVICE_INPUT_DEFAULTS("terminal", terminal)
MACHINE_CONFIG_END

/* ROM definition */
ROM_START( pipbug )
    ROM_REGION( 0x8000, "maincpu", ROMREGION_ERASEFF )
    ROM_LOAD( "pipbug.rom", 0x0000, 0x0400, CRC(f242b93e) SHA1(f82857cc882e6b5fc9f00b20b375988024f413ff))
ROM_END

/* Driver */

/*    YEAR  NAME    PARENT  COMPAT  MACHINE    INPUT   INIT     COMPANY   FULLNAME       FLAGS */
COMP( 1979, pipbug,  0,        0,     pipbug,    pipbug, driver_device,    0,  "Signetics", "PIPBUG", MACHINE_NO_SOUND_HW )
```

<- Simplified example

Delegates

```
class MyClass {
        int i;
public:
        MyClass() : i(0) { }
        virtual ~MyClass() { }
        virtual void docount(int) { i++; }
};
typedef delegate<void(int value)> callback_delegate;
MyClass mc;
callback_delegate md = callback_delegate(FUNC(MyClass::docount), &mc);
```

Why we needed delegates?

Providing callback functionality between various objects

Late binding (resolving objects referenced in runtime)

Minimal cost (using method function pointers)

Implemented in period of using C++ 98

https://github.com/mamedev/delegates

Speed measurement

Compiler	Version	OS	Time fast delegates native (ns)	Time std::function/bind (ns)
MinGW GCC	5.3.0 x64	Windows	131547400	216178100
MinGW GCC	5.3.0 x86	Windows	131160000	285218800
Clang	3.8.0 x64	Windows	100766900	219475700
GCC	4.9.2 ARM	Linux (RasPi2)	1120924321	4146617167
GCC	5.3.1 x64	Linux	139180356	205068909
Clang	3.7.0 x64	Linux	140548960	182060144
Clang Apple	7.3.0 x64	OSX	125145702	262906798
GCC	5.3.1 ARM64	Linux (Odroid-C2)	654185671	1370827564
GCC	4.9.2 MIPSEL	Linux (Creator Ci20)	1002793705	3341533518

3rd party libraries and tools

BGFX – Branimir Karadžić

LUA – PUC Rio

RapidJSON – Milo Yip

GLM – GL Math

BGFX (1/2)

Cross-platform, graphics API agnostic, "Bring Your Own Engine/Framework" style rendering library.

Supported rendering backends:

Direct3D 9

Direct3D 11

Direct3D 12 (WIP)

Metal (WIP)

OpenGL 2.1

OpenGL 3.1+

OpenGL ES 2

OpenGL ES 3.1

WebGL 1.0

WebGL 2.0

BGFX (2/2)

Supported platforms:

Android (14+, ARM, x86, MIPS)

asm.js/Emscripten (1.25.0)

FreeBSD

iOS (iPhone, iPad, AppleTV)

Linux

MIPS Creator CI20

Native Client (PPAPI 37+, ARM, x86, x64, PNaCl)

OSX (10.9+)

RaspberryPi

SteamLink

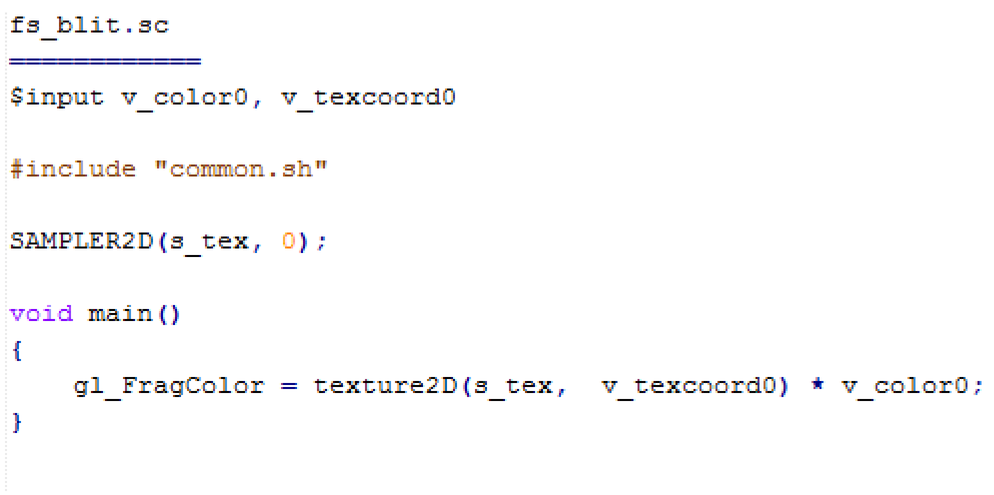
Windows (XP, Vista, 7, 8, 10)

WinRT (WinPhone 8.0+)

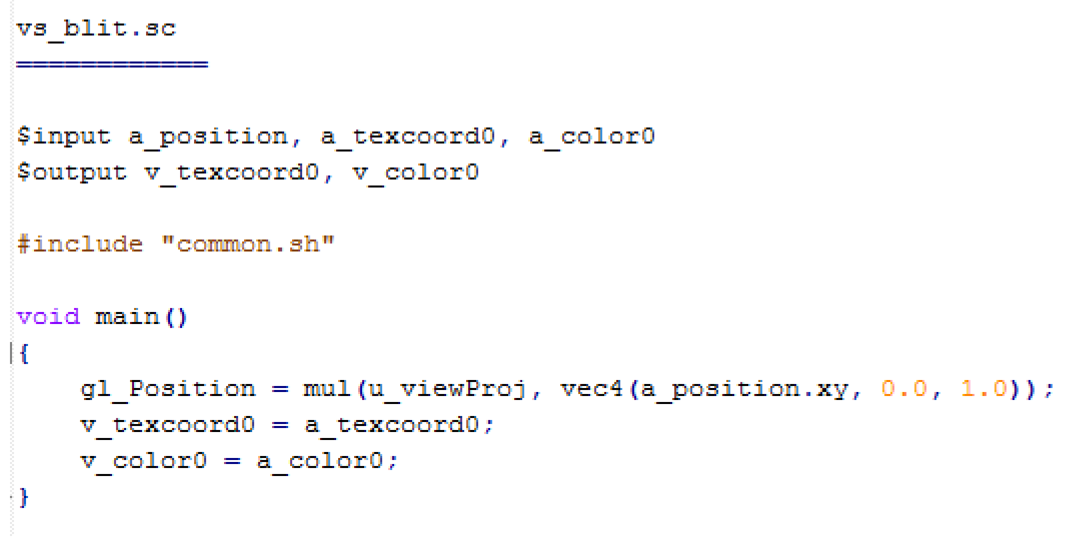
Supported compilers:

Clang 3.3 and above
GCC 4.6 and above
vs2008 and above

How do BGFX files look like ?



```
fs_blit.sc
============
$input v_color0, v_texcoord0

#include "common.sh"

SAMPLER2D(s_tex, 0);

void main()
{
    gl_FragColor = texture2D(s_tex,  v_texcoord0) * v_color0;
}
```



```
vs_blit.sc
============

$input a_position, a_texcoord0, a_color0
$output v_texcoord0, v_color0

#include "common.sh"

void main()
{
    gl_Position = mul(u_viewProj, vec4(a_position.xy, 0.0, 1.0));
    v_texcoord0 = a_texcoord0;
    v_color0 = a_color0;
}
```

Make your code public

Better feedback from users

Commits become better since people are aware more are looking at their work

More people get interested in project -> more pull requests

Do not use private repository sites to distribute your code

GIT over SVN

Do regular releases (we do it each last Wednesday of month)

https://github.com/mamedev/mame

Why join open source project?

Share your ideas

Experiment

Improve your knowledge

Knowledge gained during work on open source projects help you do your regular work.

Meet more people, learn from them.

What do we wish to offer to C++ ?

Delegates

Input handling implementation based on delegates

Definition of math for 2D and 3D graphics

Runtime shader transpiling

Let us be your playground

DEMO

Q & A

Contact :

- mmicko@gmail.com
- Twitter : https://twitter.com/micko_mame
- GitHub : https://github.com/mamedev/mame