

Porting The Mozart Virtual Machine to the Symbian Platform

Sameh El-Ansary

Konstantin Popov

Per Brand

SICS

The Oz Language

- Oz is a multi-paradigm language : It provides **object-oriented** programming, **functional** programming, **logic** programming and **constraint** programming.
- Oz allows users to dynamically create any number of sequential **dataflow threads**. A thread executing an operation will suspend until all operands needed have a well-defined value

The Mozart System

- The Mozart programming system has been developed by:
 - DFKI (the German Research Center for Artificial Intelligence),
 - SICS (the Swedish Institute of Computer Science)
 - UCL (the Université catholique de Louvain), and others.

The Target

*Make the Mozart System available
on the Symbian platform*

The Initial Picture

- Approx. **100,000 lines** of C/C++ code
- Compiled with the **GNU compiler** for all Unices and with a **cross compiler** for the win32 platforms
- Depends on the **ZLIB** library for compression
- Depends on the **GMP** library for BigNums

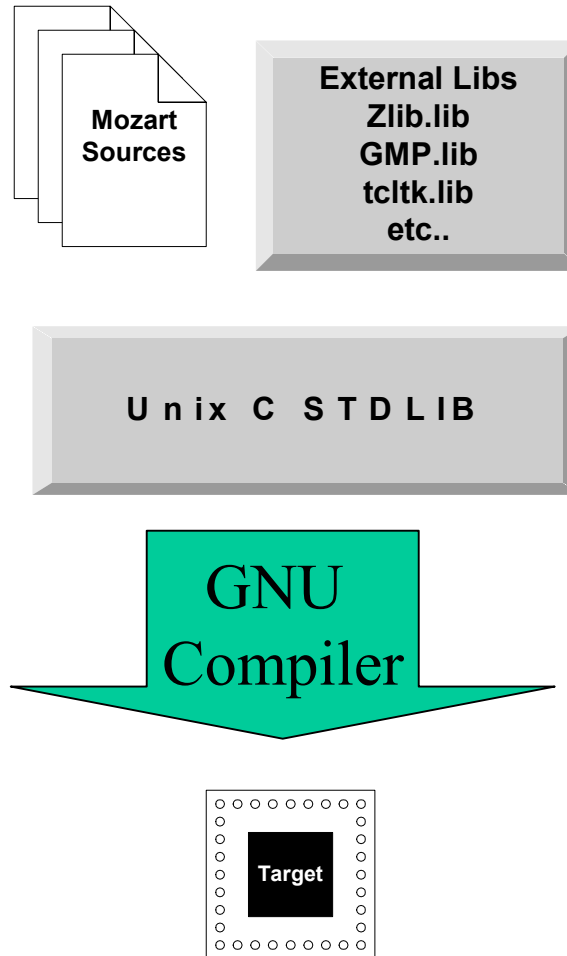
The Initial Picture –Contd.

- Extensive use of asynchronous IO through the **select()** statement
- Dependence on various kinds of OS **signals**
- Most importantly the **alarm signal** since it is used for thread scheduling

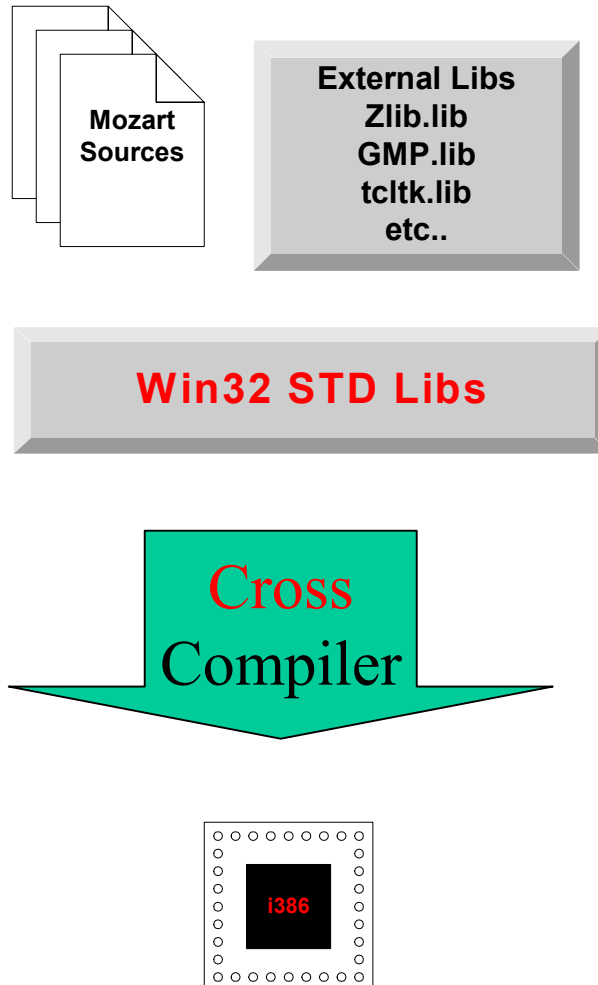
A Starting Point ?

- Rewrite it?
 - Seems a little bit too much work?
 - Smart enough?
 - Branching overhead, you are alone.
- Port it?
 - Easy? We said we have STDLIB, right?
 - No, a subset of STDLIB.

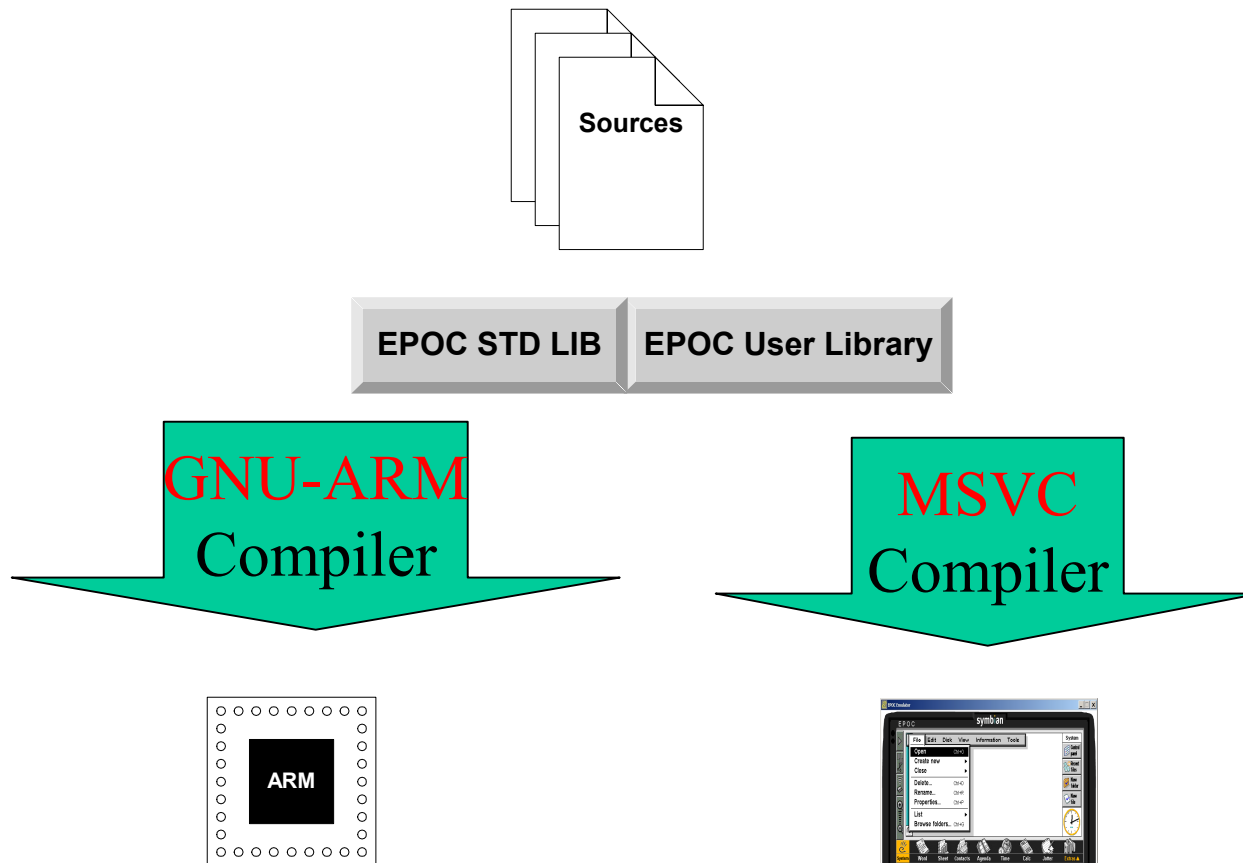
Mozart Building – UNIX platforms



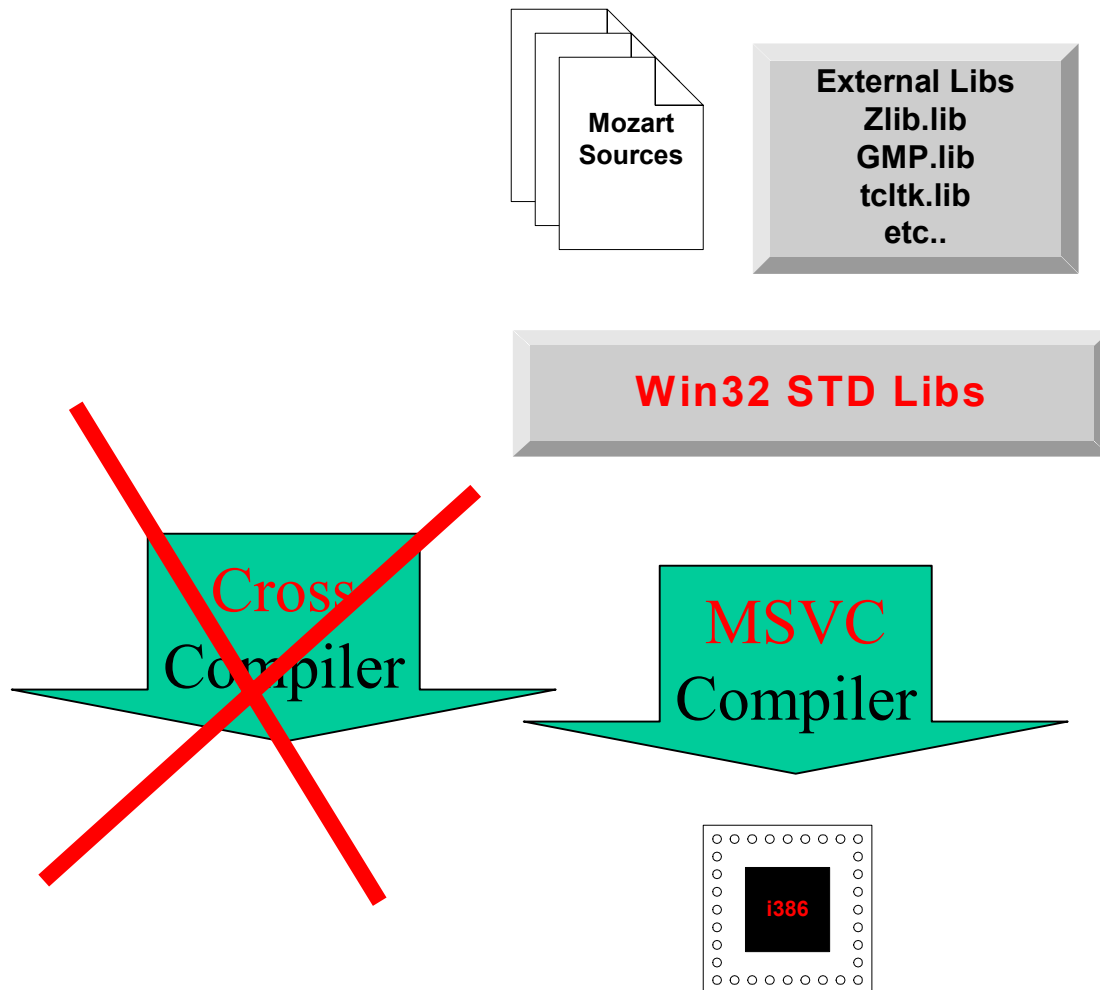
Mozart Building – Win32 platforms



General Building-Symbian



Step 1 – Compile With MSVC



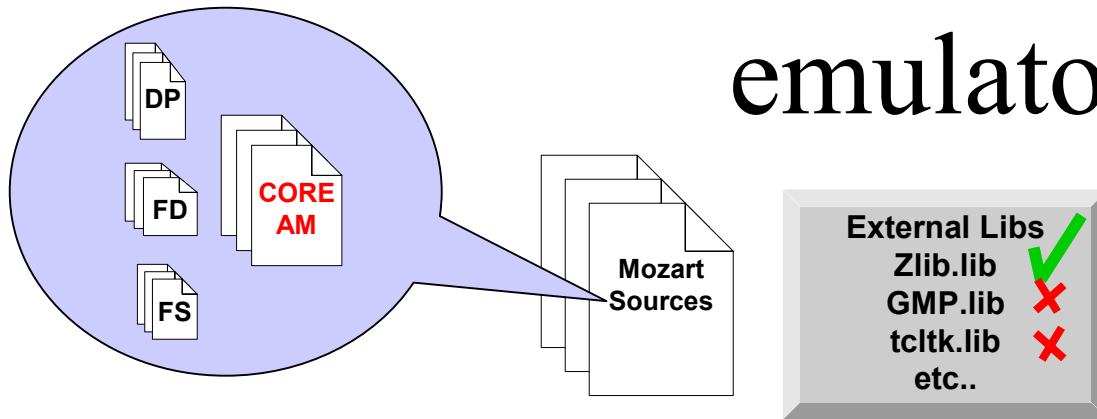
- Preparation to move to the emulator

- Included some changes:

- Minor Name clashes..

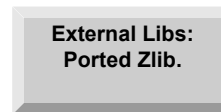
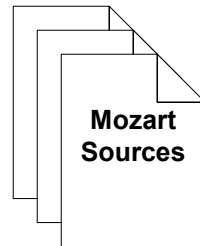
- Some casts..

Step 2 – Compile for Symbian emulator

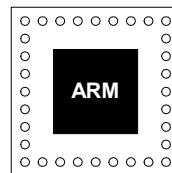


- Only Core AM
- Port ZLIB
- No GMP, No Tk
- ShutDown missing STDLIB dependencies:
No select(), No signals, ...
- A .exe not a .dll
- Memory alignment
- No stack vars > 8K
- **Result:**
Minimal functionality without threads or IO

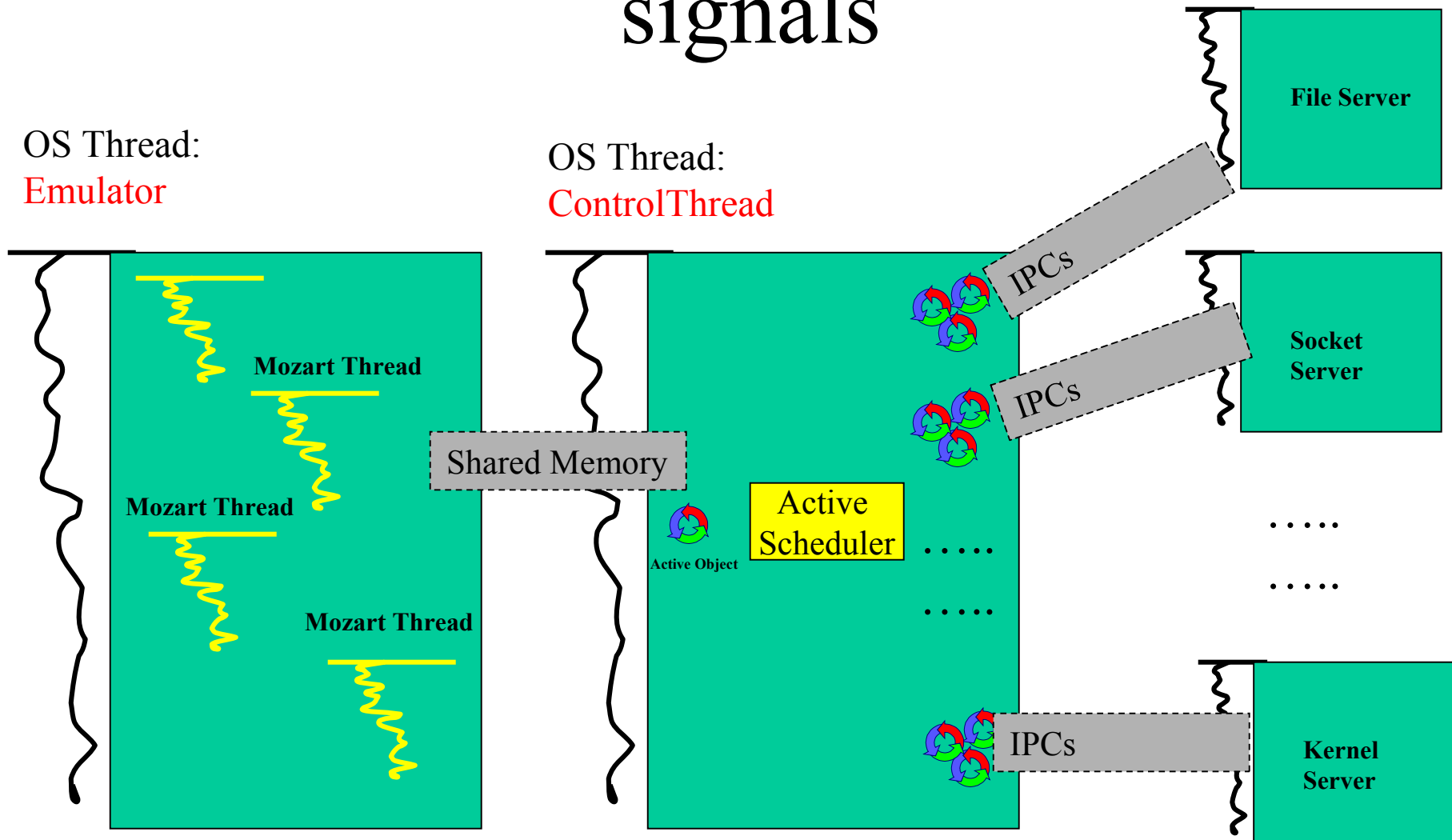
Step 3 – Compile for ARM



• Successful after some changes: Casts generate undefined symbols



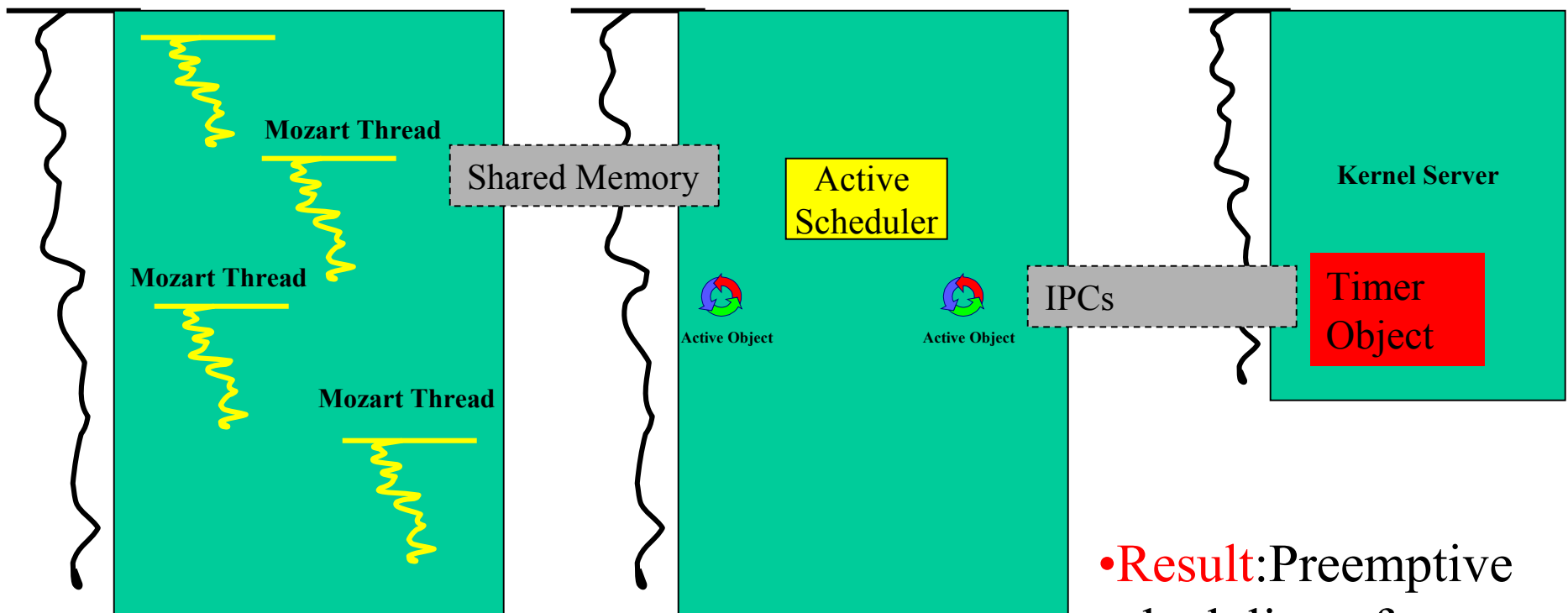
A substitute for Select() and signals



Step 4 – The Alarm signal

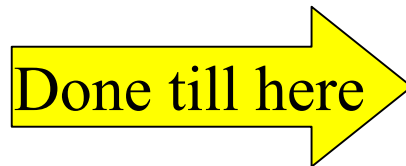
OS Thread:
Emulator

OS Thread:
ControlThread



• **Result:** Preemptive scheduling of threads

Remaining Steps



Porting Phase
Extract core Abstract Machine
Handle Differences in compilers
Handle External Libraries
Create substitutes for Async. Services
Adaptation Phase
Replacing Tk with xEikon GUI libraries
Supporting Misc Libraries: Infrared, Telephony, etc..
Minaturization/Optimization Phase
New Garbage-Collection techniques
Incremental Marshaling/Serialization
Peer Caching
....

Thank You