Embedded and Ambiens Systems (vimiac06)

Topics

1. General structure of embedded systems:typical sensors (choice of high complexity device vs. advantage / disadvantage of custom development), signal conditioning tasks, ADC and DAC types and their applications. The relationship between typical processing units (µP, µC, DSP, FPGA) in terms of device performance, design time and the complexity of the task to be solved (which task would be solved with which device).

2. Giant Gecko(EFM32-STK3700) developer card: description of typical sensors (brightness meter, LC metal sensor, touch sensor) (operating principle, what design guidelines are used, what µC peripherals are used). The principle of clock management on the Giant gecko processor is how it serves to reduce power consumption.

3. Developer environments and translators: the translation process(.c -> obj -> link). Knowledge of some typical translation interfaces (-D, -O0… -O3, -mcpu, -I, -Wall). knowledge of make program and makefile format: makefile rules (purpose, prerequisite, instruction syntax, interpretation of simpler patterns)

4. Software architectures: software architecture design aspects. Typical structure of typical software architectures (sample code, schedule diagram) and their properties: cyclic program organization (and its cases), interrupted, scheduled functions.

5. Interrupt Management: General principles for interrupt initialization and handling.General hierarchy of interrupt enable, vector interrupt handling principle, structure of the interrupt handler Cortex-M3, ATmega128 and ADSP-BF357 (in principle, no complicated diagrams are required). Some ways to specify C interrupt handling functions (Cortex-M3, ATmega128, ADSP-BF537, ADSP21364). How do functions get into the vector table, and how do we tell the compiler that they are interrupt functions?

6. Shared variables: to formulate a basic problem, for which variables it is critical, a sample example, possible solutions. Double buffering. Dynamic memory usage in embedded systems (malloc function). Stack overflow. Robust programming (timeout, secure coding, structured program organization, type usage, redundancy).

7. Troubleshooting. Debugging features of an embedded system. Tools used for debugging: debugger, tracer, profiler, watchpoint. Block diagram of a debug system built using a JTAG port, typical tasks implemented using a JTAG. Typical basic debug options (GPIO, UART: printf redirect). Methods of measuring running time.

8. SpecialC language elements: inline functions, bitfield structures, union data type, structured handling of register arrays (direct access to memory areas), attribute (eg: interrupt, always_inline, weak) and #pragma (eg: once, interrupt, align) keywords, idiom recognition . Examples.

9. Portable code: what it means, why it is important, integer data types (stdint.h), library functions (what to look for, blocking / non-blocking). Virtualization: Operating models 1 and 2, the task of the hypervisor, the requirements for it.

10. Steps in the design of data processing systems (measurement, algorithm design / testing, hardware selection, alg. implementation, testing). Data processor systems software architectures. Description of sample and block data processing software model. Sample-by-sample processing: development of latency, complexity, utilization, and determinism in the different order of timer, AD, and DA. Timing diagrams for different constructions, calculation of delay and time for data processing. Simple program code for data acquisition software (see also included in practice). Location of a data processing algorithm in the data stream. Interpretation of data processing programs. Block data processing: typical tasks and their priority (data collection and processing). Buffers treatment way: dual buffering importance. Sample example analysis from a signal processing perspective: pitch shift algorithm (data movement, decimation, buffer size selection). Block-based comparison of sample-based processing. Switch between the two modes of signal processing.

11. FPGAs:(complex block diagrams do not need to be memorized, they help to understand) How to store a configuration. Main functions of IO blocks. Clock management: main functions of DCM, topology of clock distribution network. Hierarchical structure of FPGA: CLB→Slice→ (LUT, FF + additional logics). Ways to use fast carry propagation: addition, comparator, counter, multiplier.