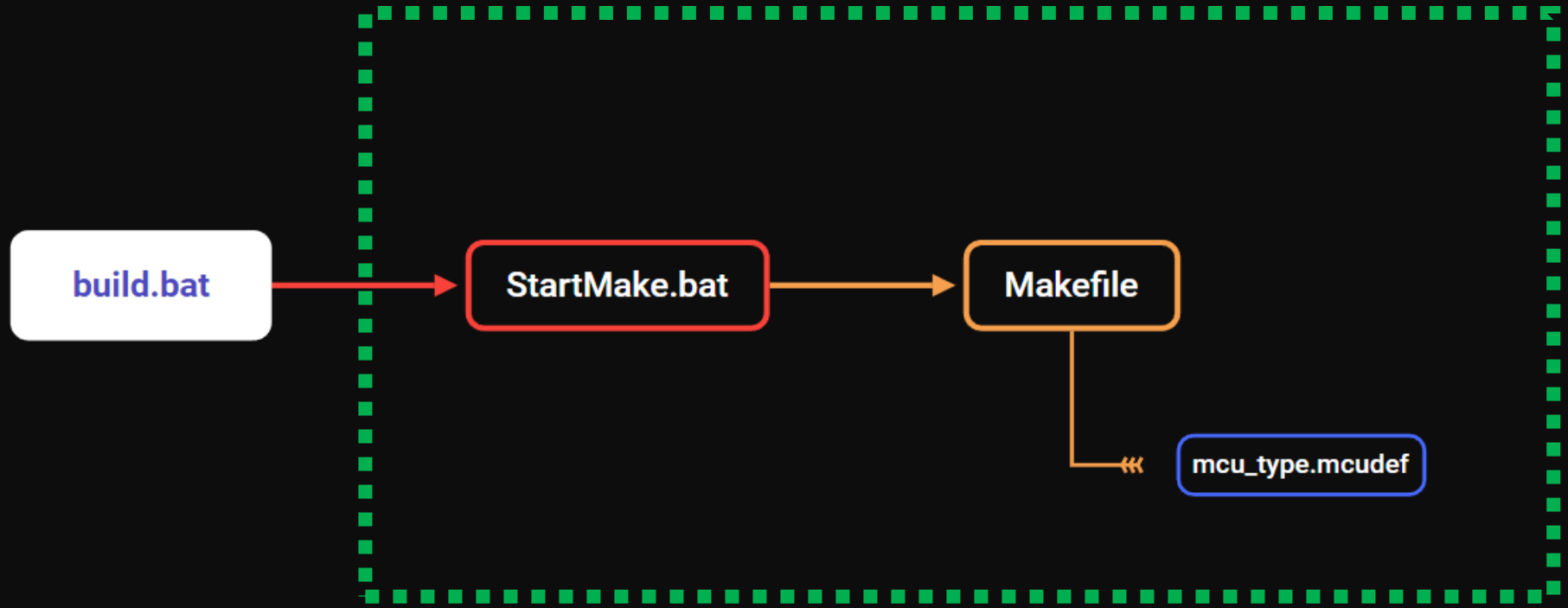
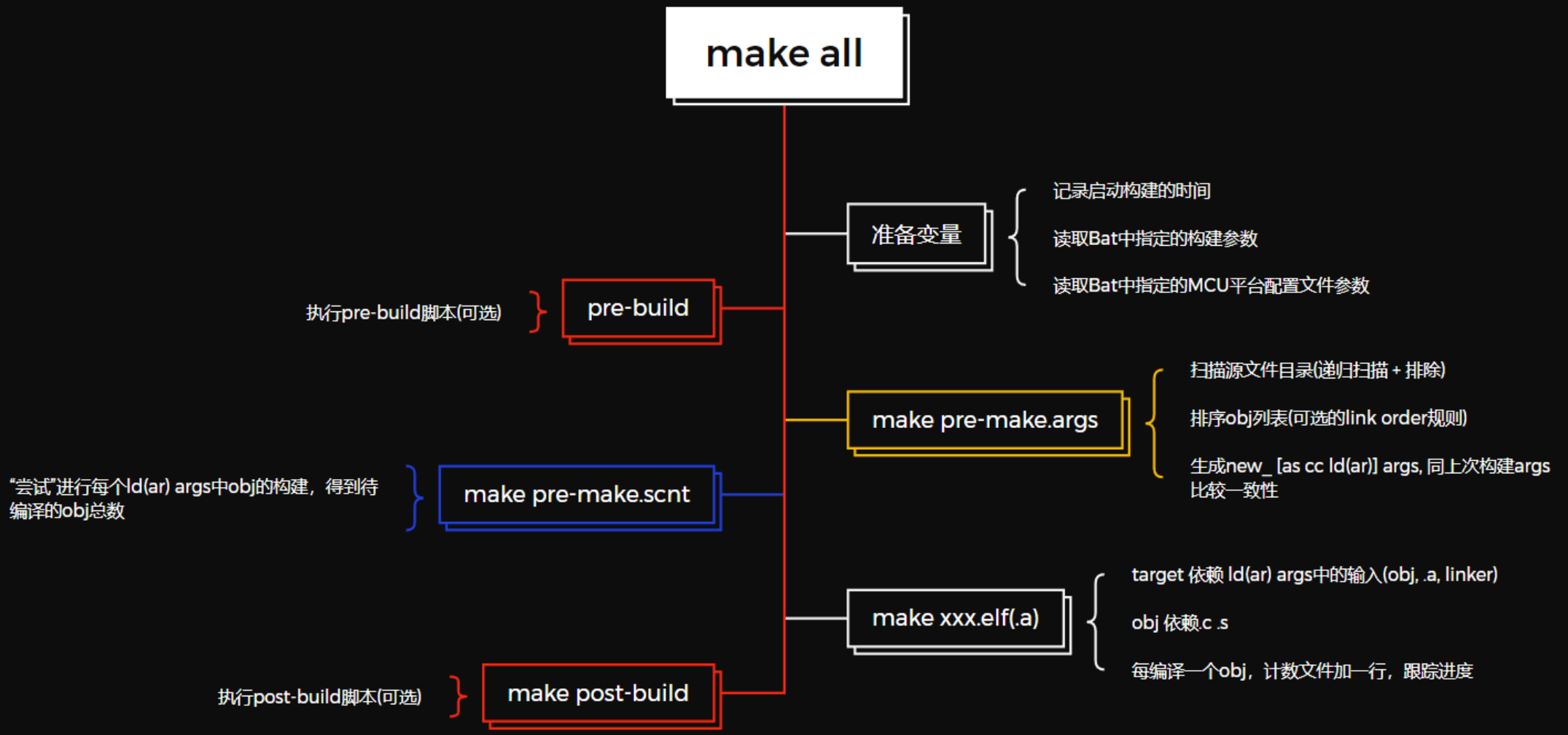
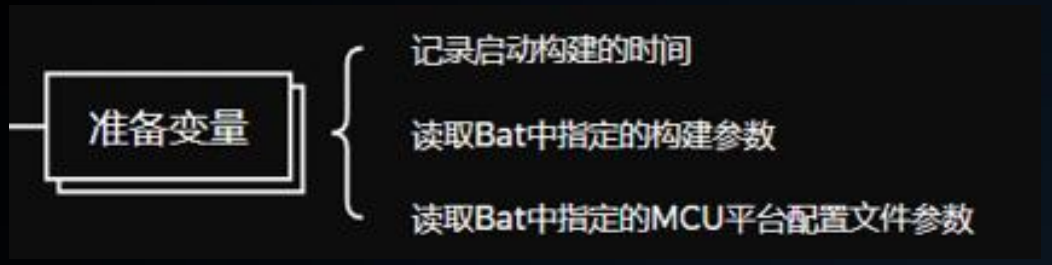
如何实现 自动化 Makefile

AutoMake

- 鱼丸ECU


build.bat
StartMake.bat
Makefile
mcu_type.mcudef


make all
准备变量
记录启动构建的时间
读取Bat中指定的构建参数
读取Bat中指定的MCU平台配置文件参数
pre-build
执行pre-build脚本(可选)
make pre-make.args
扫描源文件目录(递归扫描 + 排除)
排序obj列表(可选的link order规则)
生成new_ [as cc ld(ar)] args, 同上次构建args比较一致性
make pre-make.scnt
“尝试”进行每个ld(ar) args中obj的构建，得到待编译的obj总数
make xxx.elf(.a)
target 依赖 ld(ar) args中的输入(obj, .a, linker)
obj 依赖.c .s
每编译一个obj，计数文件加一行，跟踪进度
make post-build
执行post-build脚本(可选)



读取指定MCU平台配置

```
# tools info
LINK_ORDER := $(LINK_ORDER_$(MCU))
CC_PATH    := $(CC_PATH_$(MCU))
AR_BIN     := $(AR_BIN_$(MCU))
CC_BIN     := $(CC_BIN_$(MCU))
SZ_BIN     := $(SZ_BIN_$(MCU))
WN_BIN     := $(WN_BIN_$(MCU))

# build args
CC_ARGS    := $(CC_ARGS_$(MCU))
AS_ARGS    := $(AS_ARGS_$(MCU))
LD_ARGS    := $(LD_ARGS_$(MCU))
LIBS       := $(LIBS_$(MCU))

# binaries
AR := $(CC_PATH)/$(AR_BIN)
CC := $(CC_PATH)/$(CC_BIN)
```

时间记录:

```
START_TIME        := $(shell date +%s)
```

变量准备:
?= := = 三种变量赋值方式

```
# Variables set in environment
FORCE_TTY  ?= 0
# Variables set in command line
BUILD_PATH := build/
SRC        := ./
SRC_OUT    :=
SRC_ADD    :=
LD_FILE    :=
DST_PATH   :=
DST_JSON   :=
MCU        := s32k14x
TARGET     := default.elf
```

定义一些宏函数/变量函数

```
define PROGRESS
$$(($(1) * 100 / $(2)))% ($(1)/$(2))
endef
```

```
rwildcard           = $(foreach d, $(wildcard $1), $(call rwildcard, $d/*, $2) $(filter $2, $d))
reverse             = $(if $(word 2,$1),$(call reverse,$(wordlist 2,$(words $1),$1)) $(firstword $1),$1)
```

```
$(call PROGRESS,
```

make pre-make.args

扫描源文件目录(递归扫描 + 排除)

排序obj列表(可选的link order规则)

生成new_ [as cc ld(ar)] args, 同上次构建args比较一致性

```
# auto scan source files
$(info [Scanning sources]: out source files ...)
ifneq (, $(SRC_OUT_TRIM))
    OUT_SRCS := $(call rwildcard, $(SRC_OUT_TRIM), %.h %.H %.s %.S %.c %.C %.a %.A %.ld)
    ifneq (, $(SRC_ADD_TRIM))
        ADD_SRCS := $(call rwildcard, $(SRC_ADD_TRIM), %.h %.H %.s %.S %.c %.C %.a %.A %.ld)
        OUT_SRCS := $(filter-out $(ADD_SRCS), $(OUT_SRCS))
    endif
else
    OUT_SRCS :=
endif

$(info [Scanning sources]: .h source files ...)
H_PATH := $(foreach src, $(SRC_TRIM), $(sort $(dir $(filter-out $(OUT_SRCS), $(call rwildcard, $(src
$(info [Scanning sources]: .s source files ...)
S_SRCS := $(filter-out $(OUT_SRCS), $(call rwildcard, $(SRC_TRIM), %.s %.S))
$(info [Scanning sources]: .c source files ...)
C_SRCS := $(filter-out $(OUT_SRCS), $(call rwildcard, $(SRC_TRIM), %.c %.C))
$(info [Scanning sources]: .a source files ...)
A_SRCS := $(filter-out $(OUT_SRCS), $(call rwildcard, $(SRC_TRIM), %.a %.A))

# 若不提供LD_FILE则自动查找*flash.ld(兼容原设计)
ifneq (, $(LD_FILE_TRIM))
    LD_SRCS := $(LD_FILE_TRIM)
else
    $(info [Scanning sources]: .ld source files ...)
    LD_SRCS := $(filter-out $(OUT_SRCS), $(call rwildcard, $(SRC_TRIM), %flash.ld))
endif
```

make pre-make.args

- 扫描源文件目录(递归扫描 + 排除)
- 排序obj列表(可选的link order规则)
- 生成new_ [as cc ld(ar)] args, 同上次构建args比较一致性

```
ifeq ($(LINK_ORDER), GNU)
    OBJECTS := $(call reverse, $(sort $(dir $(S_SRCS) $(C_SRCS))))
    OBJECTS := $(foreach d, $(OBJECTS), $(sort $(wildcard $d*.[sScC])))
    OBJECTS := $(filter $(S_SRCS) $(C_SRCS), $(OBJECTS))
    OBJECTS := $(patsubst %.s,%.o,$(patsubst %.S,%.o,$(OBJECTS)))
    OBJECTS := $(patsubst %.c,%.o,$(patsubst %.C,%.o,$(OBJECTS)))
else
    OBJECTS := $(patsubst %.s,%.o,$(patsubst %.S,%.o,$(S_SRCS))) $(patsubst %.c,%.o,$(patsubst %.C,%
    OBJECTS := $(patsubst %0o,%.o,$(sort $(patsubst %.o,%0o,$(OBJECTS))))
endif
```

make pre-make.args

扫描源文件目录(递归扫描 + 排除)

排序obj列表(可选的link order规则)

生成new_ [as cc ld(ar)] args, 同上次构建args比较一致性

```
pre-make.args: $(patsubst %,_new_args_%.txt.phony,ar ld cc as)
	@echo '[Finished pre-make.args]'
	@echo ' '
```

```
_new_args_%.txt.phony: $(BUILD_DIR)/_new_args_%.txt
	@if [ -f $(BUILD_DIR)/_args_$*.txt ] && ! cmp -s $(BUILD_DIR)/_args_$*.txt $<; then \
		rm $(BUILD_DIR)/_args_$*.txt; \
		echo '[Finished checking]: $(BUILD_DIR)/_args_$*.txt is out of date and has been removed'; \
	else \
		echo '[Finished checking]: $(BUILD_DIR)/_args_$*.txt is up to date or does not exist'; \
	fi;
	@rm $<
```

关于 as cc ld args

```
$(BUILD_DIR)/%.o: %.c $(BUILD_DIR)/_args_cc.txt
ifeq ($(MAKECMDGOALS), pre-make.scnt)
    $(file >> $(BUILD_DIR)/_scnt_total.txt,$<)
else
    @echo "[CC $(call PROGRESS,$$(($(words $(file < $(BUILD_DIR)/_scnt_build.txt)) + 1)),$(SCNT_TOTAL))]: $<"
    $(file >> $(BUILD_DIR)/_scnt_build.txt,$<)
    @mkdir -p $(@D)
    @$(CC) "@$(BUILD_DIR)/_args_cc.txt" -MD -o "$@" "$(CURDIR_WIN)/$<" $(LOG_STDERR)
endif
```

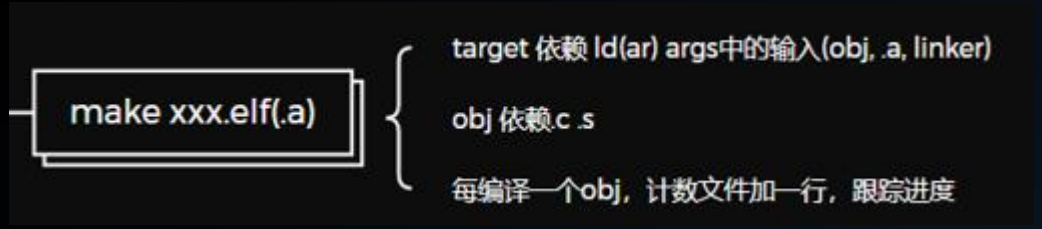
```
$(BUILD_DIR)/%args_cc.txt:
    $(file > $@,$(subst $(SPACE),$(NEWLINE),$(CC_ARGS)))
ifneq (, $(strip $(EXTRA_CC_ARGS)))
    $(file >> $@,$(subst $(SPACE),$(NEWLINE),$(strip $(EXTRA_CC_ARGS))))
endif
    $(file >> $@,$(subst $(SPACE),$(NEWLINE),$(I_PATH)))
    @echo '[Finished building]: $@'
```

"尝试"进行每个ld(ar) args中obj的构建，得到待编译的obj总数

make pre-make.scnt

```
pre-make.scnt: $(addprefix $(BUILD_DIR)/, $(OBJECTS))
	@echo '[Finished pre-make.scnt]: $(words $(file < $(BUILD_DIR)/_scnt_total.txt)) files need to be compil
	@echo ' '
```

```
$(BUILD_DIR)/%.o: %.c $(BUILD_DIR)/_args_cc.txt
ifeq ($(MAKECMDGOALS), pre-make.scnt)
	$(file >> $(BUILD_DIR)/_scnt_total.txt,$<)
else
	@echo "[CC $(call PROGRESS,$$(($(words $(file < $(BUILD_DIR)/_scnt_build.txt)) + 1)),$(SCNT_TOTAL))]: $<"
	$(file >> $(BUILD_DIR)/_scnt_build.txt,$<)
	@mkdir -p $(@D)
	@$(CC) "@$(BUILD_DIR)/_args_cc.txt" -MD -o "$@" "$(CURDIR_WIN)/$<" $(LOG_STDERR)
endif
```




```
$(BUILD_DIR)/$(TARGET_BASENAME).elf: $(BUILD_DIR)/_args_ld.txt $(addprefix $(BUILD_DIR)/, $(OBJECTS)) $(LD_SRCS) $(USER_LIBS)
    @echo ''
    @echo '[LD]: $@'
    @$(CC) -o $@ "@$<" $(LOG_STDERR)
    @echo '[Finished building target]: $@'
    @echo ''
    $(MAKE) -f $(THIS_MAKEFILE) --no-print-directory post-build
```

```
$(BUILD_DIR)/%.o: %.c $(BUILD_DIR)/_args_cc.txt
ifeq ($(MAKECMDGOALS), pre-make.scnt)
    $(file >> $(BUILD_DIR)/_scnt_total.txt,$<)
else
    @echo "[CC $(call PROGRESS,$$(($(words $(file < $(BUILD_DIR)/_scnt_build.txt)) + 1)),$(SCNT_TOTAL))]: $<"
    $(file >> $(BUILD_DIR)/_scnt_build.txt,$<)
    @mkdir -p $(@D)
    @$(CC) "@$(BUILD_DIR)/_args_cc.txt" -MD -o "$@" "$(CURDIR_WIN)/$<" $(LOG_STDERR)
endif
```