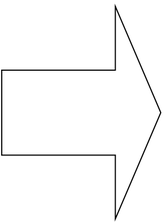

Service

BlockMessage{grpc}
+ Vg_V: double[]
+ Vm:_V double[]
+ total_duration_s: double
+ template_duration:_s double
+ sampling_freq_hz: double

StatusMessage{grpc}
+ status: int
+ message: string

StateMessage{grpc}
+ key: enum {}

Service
+ setPlan(stream BlockMessage): StatusMessage
+ getPlan(): stream BlockMessage
+ setConfig(ConfigMessage): StatusMessage
+ getConfig():ConfigMessage
+ executePlan(stream ActionMessage): stream DataMessage
+ getState(): StateMessage
+ getData(FrameDescr): stream DataMessage

FrameDecr{grpc}
+ ts:SignalTimestamp
+ size: int32

ConfigMessage{grpc}
+ config: {}
+ json_value: string

DataMessage{grpc}
+ error_message: json
+ data: json

Runtime

Runtime
+ storage: Storage
+ config_manager: ConfigManager
+ plan:Plan
+ state: RuntimeState
+ device:Device
+ update(): void
+ runNext(): void
+ stop(): void
+ setPlan(Plan):void
+ getPlan():Plan
+ setConfig(json):void
+ getConfig(): json
+ getState(): State
+ getData(from_ts, size): FrameIterator

Block
+ Vg_V: double[]
+ Vm:_V double[]
+ total_duration_s: double
+ template_duration:_s double
+ sampling_freq_hz: double

Plan
+std::vector<Block>

Storage

Storage
+ data: FrameArray
+ pushBack(Frame&&): void
+ at(TimeStamp): iterator
+ reset(): bool

FrameArray
+ List[Frame]
empty(): bool
pushBack(Frame): bool
size():size_t
signal_length():size_t
at(TimeStamp): FrameIterator
at(size_t): *Frame

Frame
+ data: Dict[ChannelKey, Signal]
+ ts_start: TimeStamp
+ ts_finish: TimeStamp
+ size(): size_t
+ detachBack(size_t): Frame
+ attachBack(size_t): Frame
+empty(): bool
+setTs(TimeStamp): void
+getTs(): TimeStamp
+keys(): List[ChannelKey]
+at(ChannelKey): Signal

FrameIterator

Signal

Device

Device
+ setProfiles(ProfilesSetup s, TimeUnit t):void
+ setDuration(TimeUnit t)
+ setReadingSampling(TimeUnit t):void
+ setTimer(Timer timer):void
+ prepare():void
+ run():void
+ stop():void
+ getData(): Frame

Timer
+ start(): bool
+ stop(): bool
+ unitsToMilliseconds(TimeUnit t): double
+ millisecondsToUnits(double):TimeUnit t
+ millisecondsToUnits(double):TimeUnit t
+ getStamp(): TimeStamp t

ProfilesSetup
+ map<ControlKey, vector<MilliVolt>>

Common

TimeStamp
+ TickCount counts
+ TimeUnit step
+ TimeStamp(tmin, tmax, fraction)
+ operator >,=,<

ControlKey
+ key: Vm , Vg

SignalKey
+ key: Uref, Utpl, Umod, Uhtr, Uaux

UcalException
+ what()

ConfigManager
+ config_addr_: string
+ write(ConfigStringKey, string): bool
+ write(ConfigDoubleKey, double):bool
+ readDouble(ConfigDoubleKey):double
+ readStr(ConfigStringKey):string

ConfigDoubleKey

ConfigStringKey