MODULE *client_integration*

External client support for Zebra specification

The specs simulates a call to the *create_account* grpc method as a starting point and then the grpc method calls the *create_account* procedure in the *zcash_client_backend* side. The grpc method then sends the key to the memory wallet and the memory wallet adds the key to the accounts set. The memory wallet then sends a block to the memory wallet and the memory wallet adds the block to the blocks set.

The memory wallet is a simple algorithm that listens for requests and sends adding requests to the scan task. The scan task listens for requests from the services process and adds tasks to the scan task set. The scan task also adds account to the memory wallet and either sends "scanned" blocks to the memory wallet or does nothing more.

The main process is the entry point of the model and calls the *create_account* grpc method.

EXTENDS TLC, $Integers$, $Sequences$, $Json$, $FiniteSets$

$StatusWaiting \triangleq$ "waiting"
$StatusAdding \triangleq$ "adding"
$CreateAccountServiceRequest \triangleq$ "create_account"

--algorithm *client_integration*

variables

A string that will be used as a response to any of the *gRPC* method calls, initially empty.

$response = $ "" **;**

The current service request flag, initially listening for requests.

$service\_request = StatusWaiting$ **;**

The current status of the scan task, initially listening for requests.

$scan\_task\_status = StatusWaiting$ **;**

The set of scan tasks that are currently being processed, initially empty.

$scan\_tasks = \{\}$ **;**

The key that will be served to the client after a create account request.

$key_to_be_served = $ "" **;**

The block that will be served to the client after a scan task finds a relevant block, initially empty.

$block\_to\_be\_served = [height \mapsto 0,\ hash \mapsto$ "000000"$]$ **;**

The set of accounts that in the memory wallet, initially empty.

$accounts = \{\}$ **;**

The set of blocks in the memory wallet, initially empty.

$blocks = \{\}$ **;**

Keep track of the last inserted accouint id.

$last\_account\_id = 0$ **;**

define

Ensure that whenever a block is available, it eventually gets inserted into the memory wallet.

$LIVENESS\_BLOCK\_INSERTION \triangleq$
$\quad \land\ block\_to\_be\_served.height > 0$
$\quad \Rightarrow \Diamond(\forall\, b \in blocks : b = block\_to\_be\_served)$

Ensure that an account is not added twice.

```
    $SAFETY\_ACCOUNT\_ADDITION \triangleq$
        $\land \forall a \in accounts :$
                $\land a.account\_id \geq 0$
                $\land \forall b \in accounts : b.account\_id \neq a.account\_id$
     Ensure that the account id is incremented properly.
    $SAFETY\_ACCOUNT\_ID\_INCREMENT \triangleq$
        $\land \forall a, b \in accounts : a.account\_id < b.account\_id$
     Ensure that a block is not inserted multiple times.
    $SAFETY\_BLOCK\_INSERTION \triangleq$
        $\land \forall b \in blocks :$
                $\land b.height > 0$
                $\land \forall c \in blocks : c.height \neq b.height$
     Ensure that the service request always return to listening after adding.
    $SERVICE\_REQUEST\_TRANSITION \triangleq$
        $\land service\_request = StatusAdding$
            $\Rightarrow \Diamond(service\_request = StatusWaiting)$
end define ;

  UTILITY PROCEDURES:

  The create_account grpc method.
procedure $create\_account\_grpc()$
begin
    $CreateAccountGrpc$:
        $service\_request := CreateAccountServiceRequest$ ;
end procedure ;

  The create_account in the zcash_client_backend side.
procedure $create\_account\_zcash\_client\_backend()$
begin
    $CreateAccountZcashClientBackend$:
        $response$ := "zxviews..." ;
        return ;
end procedure ;

  The put_block in the zcash_client_backend side.
procedure $put\_block\_zcash\_client\_backend()$
begin
    $PutBlockZcashClientBackend$:
        $blocks := blocks \cup \{block\_to\_be\_served\}$ ;
end procedure ;

  SERVICES PROCESS :

  Listen for requests and send adding requests to scan task.
process $services$ = "SERVICES"
begin
```

```
  Services:
      We only have one service request in this algorithm.
    if $service\_request = CreateAccountServiceRequest$ then
      CreateAccount:
        $scan\_task\_status := StatusAdding$ ;
      CallZcashClientBackend:
        call $create\_account\_zcash\_client\_backend()$ ;
      SendKey:
        $key\_to\_be\_served := response$ ;
    end if ;
  ServicesLoop:
    goto Services ;
end process ;
```

SCAN TASK PROCESS :

Listen for requests from the services process and :
– Add tasks to the scan task set.
– Add account to the memory wallet.
– Either send "scanned" blocks to the memory wallet or do nothing more.

```
process $scantask =$ "SCAN TASK"
variables $inner\_state = \{\}$, $inner\_accounts = \{\}$, $inner\_blocks = \{\}$, $inner\_last\_account\_id = 0$ ;
begin
  GetGlobals:
    $inner\_state := scan\_tasks$ ;
    $inner\_accounts := accounts$ ;
    $inner\_last\_account\_id := last\_account\_id$ ;
  ScanTask:
    if $scan\_task\_status = StatusAdding$ then
      AddingAccount:
        $accounts := inner\_accounts \cup \{[account\_id \mapsto last\_account\_id + 1, ufvk \mapsto key\_to\_be\_served$
        $scan\_tasks := inner\_state \cup \{key\_to\_be\_served\}$ ;
        $scan\_task\_status := StatusWaiting$ ;
        $last\_account\_id := inner\_last\_account\_id + 1$ ;
    end if ;
  SendBlock:
    either
      $block\_to\_be\_served := [height \mapsto 1, hash \mapsto$ "111111"$]$ ;
      call $put\_block\_zcash\_client\_backend()$ ;
    or
      skip ;
    end either ;
  ScanTaskLoop:
    goto ScanTask ;
end process ;
```

```
  MAIN PROCESS :

process Main = "MAIN"
begin
    CreteAccountCall:
        The grpc is the entry point of the model.
        call create_account_grpc() ;
    End:
        skip ;
end process ;

end algorithm ;
  BEGIN TRANSLATION(chksum(pcal) = "f5e7e895" ∧ chksum(tla) = "e2d0cb1f")
VARIABLES response, service_request, scan_task_status, scan_tasks,
          key_to_be_served, block_to_be_served, accounts, blocks,
          last_account_id, pc, stack

  define statement
LIVENESS_BLOCK_INSERTION ≜
    ∧ block_to_be_served.height > 0
    ⇒ ◇(∀ b ∈ blocks : b = block_to_be_served)

SAFETY_ACCOUNT_ADDITION ≜
    ∧ ∀ a ∈ accounts :
        ∧ a.account_id ≥ 0
        ∧ ∀ b ∈ accounts : b.account_id ≠ a.account_id

SAFETY_ACCOUNT_ID_INCREMENT ≜
    ∧ ∀ a, b ∈ accounts : a.account_id < b.account_id

SAFETY_BLOCK_INSERTION ≜
    ∧ ∀ b ∈ blocks :
        ∧ b.height > 0
        ∧ ∀ c ∈ blocks : c.height ≠ b.height

SERVICE_REQUEST_TRANSITION ≜
    ∧ service_request = StatusAdding
        ⇒ ◇(service_request = StatusWaiting)

VARIABLES inner_state, inner_accounts, inner_blocks, inner_last_account_id

vars ≜ ⟨response, service_request, scan_task_status, scan_tasks,
        key_to_be_served, block_to_be_served, accounts, blocks,
        last_account_id, pc, stack, inner_state, inner_accounts,
        inner_blocks, inner_last_account_id⟩

ProcSet ≜ {"SERVICES"} ∪ {"SCAN TASK"} ∪ {"MAIN"}
```

```
Init ≜  Global variables
        ∧ response = ""
        ∧ service_request = StatusWaiting
        ∧ scan_task_status = StatusWaiting
        ∧ scan_tasks = {}
        ∧ key_to_be_served = ""
        ∧ block_to_be_served = [height ↦ 0, hash ↦ "000000"]
        ∧ accounts = {}
        ∧ blocks = {}
        ∧ last_account_id = 0
         Process scantask
        ∧ inner_state = {}
        ∧ inner_accounts = {}
        ∧ inner_blocks = {}
        ∧ inner_last_account_id = 0
        ∧ stack = [self ∈ ProcSet ↦ ⟨⟩]
        ∧ pc = [self ∈ ProcSet ↦ CASE self = "SERVICES" → "Services"
                                    □ self = "SCAN TASK" → "GetGlobals"
                                    □ self = "MAIN" → "CreteAccountCall"]

CreateAccountGrpc(self) ≜ ∧ pc[self] = "CreateAccountGrpc"
                          ∧ service_request′ = CreateAccountServiceRequest
                          ∧ pc′ = [pc EXCEPT ![self] = "Error"]
                          ∧ UNCHANGED ⟨response, scan_task_status,
                                       scan_tasks, key_to_be_served,
                                       block_to_be_served, accounts,
                                       blocks, last_account_id, stack,
                                       inner_state, inner_accounts,
                                       inner_blocks, inner_last_account_id⟩

create_account_grpc(self) ≜ CreateAccountGrpc(self)

CreateAccountZcashClientBackend(self) ≜ ∧ pc[self] = "CreateAccountZcashClientBackend"
                                        ∧ response′ = "zxviews..."
                                        ∧ pc′ = [pc EXCEPT ![self] = Head(stack[self]).pc]
                                        ∧ stack′ = [stack EXCEPT ![self] = Tail(stack[self])]
                                        ∧ UNCHANGED ⟨service_request,
                                                     scan_task_status,
                                                     scan_tasks,
                                                     key_to_be_served,
                                                     block_to_be_served,
                                                     accounts, blocks,
                                                     last_account_id,
                                                     inner_state,
                                                     inner_accounts,
                                                     inner_blocks,
```

$$\mathit{inner\_last\_account\_id}\rangle$$

$$\mathit{create\_account\_zcash\_client\_backend}(\mathit{self}) \triangleq \mathit{CreateAccountZcashClientBackend}(\mathit{self})$$

$$\begin{array}{lll}
\mathit{PutBlockZcashClientBackend}(\mathit{self}) \triangleq & \land\ \mathit{pc}[\mathit{self}] = \text{“}\textsf{PutBlockZcashClientBackend}\text{”} & \\
& \land\ \mathit{blocks}' = (\mathit{blocks} \cup \{\mathit{block\_to\_be\_served}\}) & \\
& \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\mathit{self}] = \text{“}\textsf{Error}\text{”}] & \\
& \land\ \textsc{unchanged}\ \langle \mathit{response}, \mathit{service\_request}, & \\
& \qquad\qquad \mathit{scan\_task\_status}, & \\
& \qquad\qquad \mathit{scan\_tasks}, & \\
& \qquad\qquad \mathit{key\_to\_be\_served}, & \\
& \qquad\qquad \mathit{block\_to\_be\_served}, & \\
& \qquad\qquad \mathit{accounts}, \mathit{last\_account\_id}, & \\
& \qquad\qquad \mathit{stack}, \mathit{inner\_state}, & \\
& \qquad\qquad \mathit{inner\_accounts}, & \\
& \qquad\qquad \mathit{inner\_blocks}, & \\
& \qquad\qquad \mathit{inner\_last\_account\_id}\rangle &
\end{array}$$

$$\mathit{put\_block\_zcash\_client\_backend}(\mathit{self}) \triangleq \mathit{PutBlockZcashClientBackend}(\mathit{self})$$

$$\begin{array}{ll}
\mathit{Services} \triangleq & \land\ \mathit{pc}[\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{Services}\text{”} \\
& \land\ \textsc{if}\ \mathit{service\_request} = \mathit{CreateAccountServiceRequest} \\
& \qquad \textsc{then}\ \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{CreateAccount}\text{”}] \\
& \qquad \textsc{else}\ \ \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{ServicesLoop}\text{”}] \\
& \land\ \textsc{unchanged}\ \langle \mathit{response}, \mathit{service\_request}, \mathit{scan\_task\_status}, \\
& \qquad\qquad \mathit{scan\_tasks}, \mathit{key\_to\_be\_served}, \mathit{block\_to\_be\_served}, \\
& \qquad\qquad \mathit{accounts}, \mathit{blocks}, \mathit{last\_account\_id}, \mathit{stack}, \\
& \qquad\qquad \mathit{inner\_state}, \mathit{inner\_accounts}, \mathit{inner\_blocks}, \\
& \qquad\qquad \mathit{inner\_last\_account\_id}\rangle
\end{array}$$

$$\begin{array}{ll}
\mathit{CreateAccount} \triangleq & \land\ \mathit{pc}[\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{CreateAccount}\text{”} \\
& \land\ \mathit{scan\_task\_status}' = \mathit{StatusAdding} \\
& \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{CallZcashClientBackend}\text{”}] \\
& \land\ \textsc{unchanged}\ \langle \mathit{response}, \mathit{service\_request}, \mathit{scan\_tasks}, \\
& \qquad\qquad \mathit{key\_to\_be\_served}, \mathit{block\_to\_be\_served}, \\
& \qquad\qquad \mathit{accounts}, \mathit{blocks}, \mathit{last\_account\_id}, \mathit{stack}, \\
& \qquad\qquad \mathit{inner\_state}, \mathit{inner\_accounts}, \mathit{inner\_blocks}, \\
& \qquad\qquad \mathit{inner\_last\_account\_id}\rangle
\end{array}$$

$$\begin{array}{ll}
\mathit{CallZcashClientBackend} \triangleq & \land\ \mathit{pc}[\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{CallZcashClientBackend}\text{”} \\
& \land\ \mathit{stack}' = [\mathit{stack}\ \textsc{except}\ ![\text{“}\textsf{SERVICES}\text{”}] = \langle[\mathit{procedure} \mapsto \text{“}\textsf{create\_account\_z} \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ \ \mathit{pc} \qquad\quad \mapsto \text{“}\textsf{SendKey}\text{”}]\rangle \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ\ \mathit{stack}[\text{“}\textsf{SERVICES}\text{”}]] \\
& \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“}\textsf{SERVICES}\text{”}] = \text{“}\textsf{CreateAccountZcashClientBackend}\text{”}] \\
& \land\ \textsc{unchanged}\ \langle \mathit{response}, \mathit{service\_request}, \\
& \qquad\qquad \mathit{scan\_task\_status}, \mathit{scan\_tasks}, \\
& \qquad\qquad \mathit{key\_to\_be\_served}, \mathit{block\_to\_be\_served},
\end{array}$$

$$\begin{array}{l}
\qquad\qquad\qquad\qquad \mathit{accounts},\ \mathit{blocks},\ \mathit{last\_account\_id},\\
\qquad\qquad\qquad\qquad \mathit{inner\_state},\ \mathit{inner\_accounts},\\
\qquad\qquad\qquad\qquad \mathit{inner\_blocks},\ \mathit{inner\_last\_account\_id}\rangle
\end{array}$$

$$\mathit{SendKey} \triangleq \begin{array}[t]{l}
\land\ \mathit{pc}[\text{“SERVICES”}] = \text{“SendKey”}\\
\land\ \mathit{key\_to\_be\_served}' = \mathit{response}\\
\land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“SERVICES”}] = \text{“ServicesLoop”}]\\
\land\ \textsc{unchanged}\ \langle \begin{array}[t]{l}\mathit{response},\ \mathit{service\_request},\ \mathit{scan\_task\_status},\\
\mathit{scan\_tasks},\ \mathit{block\_to\_be\_served},\ \mathit{accounts},\ \mathit{blocks},\\
\mathit{last\_account\_id},\ \mathit{stack},\ \mathit{inner\_state},\ \mathit{inner\_accounts},\\
\mathit{inner\_blocks},\ \mathit{inner\_last\_account\_id}\rangle\end{array}
\end{array}$$

$$\mathit{ServicesLoop} \triangleq \begin{array}[t]{l}
\land\ \mathit{pc}[\text{“SERVICES”}] = \text{“ServicesLoop”}\\
\land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“SERVICES”}] = \text{“Services”}]\\
\land\ \textsc{unchanged}\ \langle \begin{array}[t]{l}\mathit{response},\ \mathit{service\_request},\ \mathit{scan\_task\_status},\\
\mathit{scan\_tasks},\ \mathit{key\_to\_be\_served},\\
\mathit{block\_to\_be\_served},\ \mathit{accounts},\ \mathit{blocks},\\
\mathit{last\_account\_id},\ \mathit{stack},\ \mathit{inner\_state},\\
\mathit{inner\_accounts},\ \mathit{inner\_blocks},\\
\mathit{inner\_last\_account\_id}\rangle\end{array}
\end{array}$$

$$\mathit{services} \triangleq \begin{array}[t]{l}\mathit{Services} \lor \mathit{CreateAccount} \lor \mathit{CallZcashClientBackend} \lor \mathit{SendKey}\\
\quad \lor\ \mathit{ServicesLoop}\end{array}$$

$$\mathit{GetGlobals} \triangleq \begin{array}[t]{l}
\land\ \mathit{pc}[\text{“SCAN TASK”}] = \text{“GetGlobals”}\\
\land\ \mathit{inner\_state}' = \mathit{scan\_tasks}\\
\land\ \mathit{inner\_accounts}' = \mathit{accounts}\\
\land\ \mathit{inner\_last\_account\_id}' = \mathit{last\_account\_id}\\
\land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“SCAN TASK”}] = \text{“ScanTask”}]\\
\land\ \textsc{unchanged}\ \langle \begin{array}[t]{l}\mathit{response},\ \mathit{service\_request},\ \mathit{scan\_task\_status},\\
\mathit{scan\_tasks},\ \mathit{key\_to\_be\_served},\ \mathit{block\_to\_be\_served},\\
\mathit{accounts},\ \mathit{blocks},\ \mathit{last\_account\_id},\ \mathit{stack},\\
\mathit{inner\_blocks}\rangle\end{array}
\end{array}$$

$$\mathit{ScanTask} \triangleq \begin{array}[t]{l}
\land\ \mathit{pc}[\text{“SCAN TASK”}] = \text{“ScanTask”}\\
\land\ \textsc{if}\ \mathit{scan\_task\_status} = \mathit{StatusAdding}\\
\qquad \textsc{then}\ \ \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“SCAN TASK”}] = \text{“AddingAccount”}]\\
\qquad \textsc{else}\ \ \ \land\ \mathit{pc}' = [\mathit{pc}\ \textsc{except}\ ![\text{“SCAN TASK”}] = \text{“SendBlock”}]\\
\land\ \textsc{unchanged}\ \langle \begin{array}[t]{l}\mathit{response},\ \mathit{service\_request},\ \mathit{scan\_task\_status},\\
\mathit{scan\_tasks},\ \mathit{key\_to\_be\_served},\ \mathit{block\_to\_be\_served},\\
\mathit{accounts},\ \mathit{blocks},\ \mathit{last\_account\_id},\ \mathit{stack},\\
\mathit{inner\_state},\ \mathit{inner\_accounts},\ \mathit{inner\_blocks},\\
\mathit{inner\_last\_account\_id}\rangle\end{array}
\end{array}$$

$$\mathit{AddingAccount} \triangleq \begin{array}[t]{l}
\land\ \mathit{pc}[\text{“SCAN TASK”}] = \text{“AddingAccount”}\\
\land\ \mathit{accounts}' = (\mathit{inner\_accounts} \cup \{[\mathit{account\_id} \mapsto \mathit{last\_account\_id} + 1,\ \mathit{ufvk} \mapsto \mathit{key\_to\_b}\\
\land\ \mathit{scan\_tasks}' = (\mathit{inner\_state} \cup \{\mathit{key\_to\_be\_served}\})
\end{array}$$

$$
\begin{array}{l}
\land\ \mathit{scan\_task\_status}' = \mathit{StatusWaiting} \\
\land\ \mathit{last\_account\_id}' = \mathit{inner\_last\_account\_id} + 1 \\
\land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“SendBlock”}] \\
\land\ \text{UNCHANGED } \langle \mathit{response}, \mathit{service\_request}, \mathit{key\_to\_be\_served}, \\
\qquad\qquad\qquad \mathit{block\_to\_be\_served}, \mathit{blocks}, \mathit{stack}, \\
\qquad\qquad\qquad \mathit{inner\_state}, \mathit{inner\_accounts}, \mathit{inner\_blocks}, \\
\qquad\qquad\qquad \mathit{inner\_last\_account\_id} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{SendBlock} & \triangleq & \land\ \mathit{pc}[\text{“SCAN TASK”}] = \text{“SendBlock”} \\
& & \land\ \lor\ \land\ \mathit{block\_to\_be\_served}' = [\mathit{height} \mapsto 1, \mathit{hash} \mapsto \text{“111111”}] \\
& & \qquad\ \land\ \mathit{stack}' = [\mathit{stack} \text{ EXCEPT } ![\text{“SCAN TASK”}] = \langle [\mathit{procedure} \mapsto \text{“put\_block\_zcash\_client\_} \\
& & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\ \ \mathit{pc} \qquad\quad\ \mapsto \text{“ScanTaskLoop”}] \rangle \\
& & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\ \ \circ\ \mathit{stack}[\text{“SCAN TASK”}]] \\
& & \qquad\ \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“PutBlockZcashClientBackend”}] \\
& & \quad\ \lor\ \land\ \text{TRUE} \\
& & \qquad\ \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“ScanTaskLoop”}] \\
& & \qquad\ \land\ \text{UNCHANGED } \langle \mathit{block\_to\_be\_served}, \mathit{stack} \rangle \\
& & \land\ \text{UNCHANGED } \langle \mathit{response}, \mathit{service\_request}, \mathit{scan\_task\_status}, \\
& & \qquad\qquad\qquad\quad \mathit{scan\_tasks}, \mathit{key\_to\_be\_served}, \mathit{accounts}, \mathit{blocks}, \\
& & \qquad\qquad\qquad\quad \mathit{last\_account\_id}, \mathit{inner\_state}, \mathit{inner\_accounts}, \\
& & \qquad\qquad\qquad\quad \mathit{inner\_blocks}, \mathit{inner\_last\_account\_id} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{ScanTaskLoop} & \triangleq & \land\ \mathit{pc}[\text{“SCAN TASK”}] = \text{“ScanTaskLoop”} \\
& & \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“ScanTask”}] \\
& & \land\ \text{UNCHANGED } \langle \mathit{response}, \mathit{service\_request}, \mathit{scan\_task\_status}, \\
& & \qquad\qquad\qquad\quad \mathit{scan\_tasks}, \mathit{key\_to\_be\_served}, \\
& & \qquad\qquad\qquad\quad \mathit{block\_to\_be\_served}, \mathit{accounts}, \mathit{blocks}, \\
& & \qquad\qquad\qquad\quad \mathit{last\_account\_id}, \mathit{stack}, \mathit{inner\_state}, \\
& & \qquad\qquad\qquad\quad \mathit{inner\_accounts}, \mathit{inner\_blocks}, \\
& & \qquad\qquad\qquad\quad \mathit{inner\_last\_account\_id} \rangle
\end{array}
$$

$$
\begin{array}{lll}
\mathit{scantask} & \triangleq & \mathit{GetGlobals} \lor \mathit{ScanTask} \lor \mathit{AddingAccount} \lor \mathit{SendBlock} \\
& & \quad \lor\ \mathit{ScanTaskLoop}
\end{array}
$$

$$
\begin{array}{lll}
\mathit{CreteAccountCall} & \triangleq & \land\ \mathit{pc}[\text{“MAIN”}] = \text{“CreteAccountCall”} \\
& & \land\ \mathit{stack}' = [\mathit{stack} \text{ EXCEPT } ![\text{“MAIN”}] = \langle [\mathit{procedure} \mapsto \text{“create\_account\_grpc”}, \\
& & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\ \ \mathit{pc} \qquad\quad\ \mapsto \text{“End”}] \rangle \\
& & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\ \ \circ\ \mathit{stack}[\text{“MAIN”}]] \\
& & \land\ \mathit{pc}' = [\mathit{pc} \text{ EXCEPT } ![\text{“MAIN”}] = \text{“CreateAccountGrpc”}] \\
& & \land\ \text{UNCHANGED } \langle \mathit{response}, \mathit{service\_request}, \\
& & \qquad\qquad\qquad\quad \mathit{scan\_task\_status}, \mathit{scan\_tasks}, \\
& & \qquad\qquad\qquad\quad \mathit{key\_to\_be\_served}, \mathit{block\_to\_be\_served}, \\
& & \qquad\qquad\qquad\quad \mathit{accounts}, \mathit{blocks}, \mathit{last\_account\_id}, \\
& & \qquad\qquad\qquad\quad \mathit{inner\_state}, \mathit{inner\_accounts}, \mathit{inner\_blocks}, \\
& & \qquad\qquad\qquad\quad \mathit{inner\_last\_account\_id} \rangle
\end{array}
$$

$$\mathit{End} \triangleq \begin{array}{l} \land\, pc[\text{“MAIN”}] = \text{“End”} \\ \land\, \text{TRUE} \\ \land\, pc' = [pc \text{ EXCEPT } ![\text{“MAIN”}] = \text{“Done”}] \\ \land\, \text{UNCHANGED } \langle \mathit{response}, \mathit{service\_request}, \mathit{scan\_task\_status}, \mathit{scan\_tasks}, \\ \qquad\qquad\qquad \mathit{key\_to\_be\_served}, \mathit{block\_to\_be\_served}, \mathit{accounts}, \mathit{blocks}, \\ \qquad\qquad\qquad \mathit{last\_account\_id}, \mathit{stack}, \mathit{inner\_state}, \mathit{inner\_accounts}, \\ \qquad\qquad\qquad \mathit{inner\_blocks}, \mathit{inner\_last\_account\_id} \rangle \end{array}$$

$$\mathit{Main} \triangleq \mathit{CreteAccountCall} \lor \mathit{End}$$

Allow infinite stuttering to prevent deadlock on termination.

$$\mathit{Terminating} \triangleq \begin{array}{l} \land\, \forall\, \mathit{self} \in \mathit{ProcSet} : pc[\mathit{self}] = \text{“Done”} \\ \land\, \text{UNCHANGED } \mathit{vars} \end{array}$$

$$\mathit{Next} \triangleq \begin{array}{l} \mathit{services} \lor \mathit{scantask} \lor \mathit{Main} \\ \quad \lor\, (\exists\, \mathit{self} \in \mathit{ProcSet} : \begin{array}{l} \lor\, \mathit{create\_account\_grpc}(\mathit{self}) \\ \lor\, \mathit{create\_account\_zcash\_client\_backend}(\mathit{self}) \\ \lor\, \mathit{put\_block\_zcash\_client\_backend}(\mathit{self})) \end{array} \\ \quad \lor\, \mathit{Terminating} \end{array}$$

$$\mathit{Spec} \triangleq \mathit{Init} \land \Box[\mathit{Next}]_{\mathit{vars}}$$

$$\mathit{Termination} \triangleq \Diamond(\forall\, \mathit{self} \in \mathit{ProcSet} : pc[\mathit{self}] = \text{“Done”})$$

END TRANSLATION