MODULE *wallet_integration*

The memory wallet integration with 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 *wallet_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.
$scan\_tasks = \{\}$;

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 accounts that are currently being processed, initially empty.
$accounts = \langle [account\_id \mapsto 0,\ ufvk \mapsto ""] \rangle$;

The blocks that are currently being processed, initially empty.
$blocks = \langle\rangle$;

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 := Append(blocks, 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 = ⟨⟩ ;
begin
    GetScanTasks:
        inner_state := scan_tasks ;
    GetAccounts:
        inner_accounts := accounts ;
```

```
    GetBlocks:
        inner_blocks := blocks ;
    ScanTask:
        if scan_task_status = StatusAdding then
            AddingAccount:
                accounts := Append(inner_accounts, [account_id ↦ inner_accounts[Len(inner_accounts)].a
                inner_state := inner_state ∪ {key_to_be_served} ;
                scan_task_status := StatusWaiting ;
        end if ;
    SendBlock:
        either
            block_to_be_served := [height ↦ 1, hash ↦ "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*) = "2e1ecc63" ∧ *chksum*(*tla*) = "d770e1ae")

VARIABLES *response*, *service_request*, *scan_task_status*, *scan_tasks*, *key_to_be_served*, *block_to_be_served*, *accounts*, *blocks*, *pc*, *stack*, *inner_state*, *inner_accounts*, *inner_blocks*

$vars \triangleq \langle response, service\_request, scan\_task\_status, scan\_tasks, key\_to\_be\_served, block\_to\_be\_served, accounts, blocks, pc, stack, inner\_state, inner\_accounts, inner\_blocks \rangle$

$ProcSet \triangleq \{\text{"SERVICES"}\} \cup \{\text{"SCAN TASK"}\} \cup \{\text{"MAIN"}\}$

$Init \triangleq$ *Global variables*
$\wedge\ response = \text{""}$
$\wedge\ service\_request = StatusWaiting$
$\wedge\ scan\_task\_status = StatusWaiting$

$\land\ scan\_tasks = \{\}$
$\land\ key\_to\_be\_served = \text{“”}$
$\land\ block\_to\_be\_served = [height \mapsto 0,\ hash \mapsto \text{“000000”}]$
$\land\ accounts = \langle [account\_id \mapsto 0,\ ufvk \mapsto \text{“”}] \rangle$
$\land\ blocks = \langle \rangle$
Process scantask
$\land\ inner\_state = \{\}$
$\land\ inner\_accounts = \langle \rangle$
$\land\ inner\_blocks = \langle \rangle$
$\land\ stack = [self \in ProcSet \mapsto \langle \rangle]$
$\land\ pc = [self \in ProcSet \mapsto \text{CASE } self = \text{“SERVICES”} \to \text{“Services”}$
$\qquad\qquad\qquad\qquad\qquad\qquad \square\ \ self = \text{“SCAN TASK”} \to \text{“GetScanTasks”}$
$\qquad\qquad\qquad\qquad\qquad\qquad \square\ \ self = \text{“MAIN”} \to \text{“CreteAccountCall”}]$

$CreateAccountGrpc(self) \triangleq \land\ pc[self] = \text{“CreateAccountGrpc”}$
$\qquad\qquad\qquad\qquad\qquad\qquad \land\ service\_request' = CreateAccountServiceRequest$
$\qquad\qquad\qquad\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Error”}]$
$\qquad\qquad\qquad\qquad\qquad\qquad \land \text{UNCHANGED } \langle response,\ scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad scan\_tasks,\ key\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad block\_to\_be\_served,\ accounts,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad blocks,\ stack,\ inner\_state,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad inner\_accounts,\ inner\_blocks\rangle$

$create\_account\_grpc(self) \triangleq CreateAccountGrpc(self)$

$CreateAccountZcashClientBackend(self) \triangleq \land\ pc[self] = \text{“CreateAccountZcashClientBackend”}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ response' = \text{“zxviews...”}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land \text{UNCHANGED } \langle service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad scan\_task\_status,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad scan\_tasks,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad key\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad block\_to\_be\_served,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad accounts,\ blocks,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad inner\_state,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad inner\_accounts,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad inner\_blocks\rangle$

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

$PutBlockZcashClientBackend(self) \triangleq \land\ pc[self] = \text{“PutBlockZcashClientBackend”}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ blocks' = Append(blocks,\ block\_to\_be\_served)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Error”}]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land \text{UNCHANGED } \langle response,\ service\_request,$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad scan\_task\_status,$

$scan\_tasks$,
$key\_to\_be\_served$,
$block\_to\_be\_served$,
$accounts$, $stack$,
$inner\_state$,
$inner\_accounts$,
$inner\_blocks\rangle$

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

$$
\begin{aligned}
Services \triangleq\ & \land pc[\text{“SERVICES”}] = \text{“Services”} \\
& \land \text{IF } service\_request = CreateAccountServiceRequest \\
& \quad \text{THEN } \land pc' = [pc \text{ EXCEPT } ![\text{“SERVICES”}] = \text{“CreateAccount”}] \\
& \quad \text{ELSE } \land pc' = [pc \text{ EXCEPT } ![\text{“SERVICES”}] = \text{“ServicesLoop”}] \\
& \land \text{UNCHANGED } \langle response, service\_request, scan\_task\_status, \\
& \qquad scan\_tasks, key\_to\_be\_served, block\_to\_be\_served, \\
& \qquad accounts, blocks, stack, inner\_state, \\
& \qquad inner\_accounts, inner\_blocks\rangle
\end{aligned}
$$

$$
\begin{aligned}
CreateAccount \triangleq\ & \land pc[\text{“SERVICES”}] = \text{“CreateAccount”} \\
& \land scan\_task\_status' = StatusAdding \\
& \land pc' = [pc \text{ EXCEPT } ![\text{“SERVICES”}] = \text{“CallZcashClientBackend”}] \\
& \land \text{UNCHANGED } \langle response, service\_request, scan\_tasks, \\
& \qquad key\_to\_be\_served, block\_to\_be\_served, \\
& \qquad accounts, blocks, stack, inner\_state, \\
& \qquad inner\_accounts, inner\_blocks\rangle
\end{aligned}
$$

$$
\begin{aligned}
CallZcashClientBackend \triangleq\ & \land pc[\text{“SERVICES”}] = \text{“CallZcashClientBackend”} \\
& \land stack' = [stack \text{ EXCEPT } ![\text{“SERVICES”}] = \langle [procedure \mapsto \text{“create\_account\_z} \\
& \qquad pc \mapsto \text{“SendKey”}] \rangle \\
& \qquad \circ stack[\text{“SERVICES”}]] \\
& \land pc' = [pc \text{ EXCEPT } ![\text{“SERVICES”}] = \text{“CreateAccountZcashClientBackend”}] \\
& \land \text{UNCHANGED } \langle response, service\_request, \\
& \qquad scan\_task\_status, scan\_tasks, \\
& \qquad key\_to\_be\_served, block\_to\_be\_served, \\
& \qquad accounts, blocks, inner\_state, \\
& \qquad inner\_accounts, inner\_blocks\rangle
\end{aligned}
$$

$$
\begin{aligned}
SendKey \triangleq\ & \land pc[\text{“SERVICES”}] = \text{“SendKey”} \\
& \land key\_to\_be\_served' = response \\
& \land pc' = [pc \text{ EXCEPT } ![\text{“SERVICES”}] = \text{“ServicesLoop”}] \\
& \land \text{UNCHANGED } \langle response, service\_request, scan\_task\_status, \\
& \qquad scan\_tasks, block\_to\_be\_served, accounts, blocks, \\
& \qquad stack, inner\_state, inner\_accounts, inner\_blocks\rangle
\end{aligned}
$$

$$
\begin{aligned}
ServicesLoop \triangleq\ & \land pc[\text{“SERVICES”}] = \text{“ServicesLoop”} \\
& \land pc' = [pc \text{ EXCEPT } ![\text{“SERVICES”}] = \text{“Services”}]
\end{aligned}
$$

$\land$ UNCHANGED $\langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served,$
$block\_to\_be\_served, accounts, blocks, stack,$
$inner\_state, inner\_accounts, inner\_blocks\rangle$

$services \triangleq Services \lor CreateAccount \lor CallZcashClientBackend \lor SendKey$
$\lor ServicesLoop$

$GetScanTasks \triangleq$ $\land pc[\text{“SCAN TASK”}] = \text{“GetScanTasks”}$
$\land inner\_state' = scan\_tasks$
$\land pc' = [pc \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“GetAccounts”}]$
$\land$ UNCHANGED $\langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served,$
$block\_to\_be\_served, accounts, blocks, stack,$
$inner\_accounts, inner\_blocks\rangle$

$GetAccounts \triangleq$ $\land pc[\text{“SCAN TASK”}] = \text{“GetAccounts”}$
$\land inner\_accounts' = accounts$
$\land pc' = [pc \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“GetBlocks”}]$
$\land$ UNCHANGED $\langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served,$
$block\_to\_be\_served, accounts, blocks, stack,$
$inner\_state, inner\_blocks\rangle$

$GetBlocks \triangleq$ $\land pc[\text{“SCAN TASK”}] = \text{“GetBlocks”}$
$\land inner\_blocks' = blocks$
$\land pc' = [pc \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“ScanTask”}]$
$\land$ UNCHANGED $\langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served, block\_to\_be\_served,$
$accounts, blocks, stack, inner\_state,$
$inner\_accounts\rangle$

$ScanTask \triangleq$ $\land pc[\text{“SCAN TASK”}] = \text{“ScanTask”}$
$\land$ IF $scan\_task\_status = StatusAdding$
THEN $\land pc' = [pc \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“AddingAccount”}]$
ELSE $\land pc' = [pc \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“SendBlock”}]$
$\land$ UNCHANGED $\langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served, block\_to\_be\_served,$
$accounts, blocks, stack, inner\_state,$
$inner\_accounts, inner\_blocks\rangle$

$AddingAccount \triangleq$ $\land pc[\text{“SCAN TASK”}] = \text{“AddingAccount”}$
$\land accounts' = Append(inner\_accounts, [account\_id \mapsto inner\_accounts[Len(inner\_acco$
$\land inner\_state' = (inner\_state \cup \{key\_to\_be\_served\})$
$\land scan\_task\_status' = StatusWaiting$
$\land pc' = [pc \text{ EXCEPT } ![\text{“SCAN TASK”}] = \text{“SendBlock”}]$
$\land$ UNCHANGED $\langle response, service\_request, scan\_tasks,$

$key\_to\_be\_served$, $block\_to\_be\_served$, $blocks$,
$stack$, $inner\_accounts$, $inner\_blocks\rangle$

$SendBlock \triangleq \land pc[\text{"SCAN TASK"}] = \text{"SendBlock"}$
$\land \lor \land block\_to\_be\_served' = [height \mapsto 1, hash \mapsto \text{"111111"}]$
$\land stack' = [stack \text{ EXCEPT } ![\text{"SCAN TASK"}] = \langle[procedure \mapsto \text{"put\_block\_zcash\_client\_}$
$pc \mapsto \text{"ScanTaskLoop"}]\rangle$
$\circ stack[\text{"SCAN TASK"}]]$
$\land pc' = [pc \text{ EXCEPT } ![\text{"SCAN TASK"}] = \text{"PutBlockZcashClientBackend"}]$
$\lor \land \text{TRUE}$
$\land pc' = [pc \text{ EXCEPT } ![\text{"SCAN TASK"}] = \text{"ScanTaskLoop"}]$
$\land \text{UNCHANGED } \langle block\_to\_be\_served, stack\rangle$
$\land \text{UNCHANGED } \langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served, accounts, blocks,$
$inner\_state, inner\_accounts, inner\_blocks\rangle$

$ScanTaskLoop \triangleq \land pc[\text{"SCAN TASK"}] = \text{"ScanTaskLoop"}$
$\land pc' = [pc \text{ EXCEPT } ![\text{"SCAN TASK"}] = \text{"ScanTask"}]$
$\land \text{UNCHANGED } \langle response, service\_request, scan\_task\_status,$
$scan\_tasks, key\_to\_be\_served,$
$block\_to\_be\_served, accounts, blocks, stack,$
$inner\_state, inner\_accounts, inner\_blocks\rangle$

$scantask \triangleq GetScanTasks \lor GetAccounts \lor GetBlocks \lor ScanTask$
$\lor AddingAccount \lor SendBlock \lor ScanTaskLoop$

$CreteAccountCall \triangleq \land pc[\text{"MAIN"}] = \text{"CreteAccountCall"}$
$\land stack' = [stack \text{ EXCEPT } ![\text{"MAIN"}] = \langle[procedure \mapsto \text{"create\_account\_grpc"},$
$pc \mapsto \text{"End"}]\rangle$
$\circ stack[\text{"MAIN"}]]$
$\land pc' = [pc \text{ EXCEPT } ![\text{"MAIN"}] = \text{"CreateAccountGrpc"}]$
$\land \text{UNCHANGED } \langle response, service\_request,$
$scan\_task\_status, scan\_tasks,$
$key\_to\_be\_served, block\_to\_be\_served,$
$accounts, blocks, inner\_state,$
$inner\_accounts, inner\_blocks\rangle$

$End \triangleq \land pc[\text{"MAIN"}] = \text{"End"}$
$\land \text{TRUE}$
$\land pc' = [pc \text{ EXCEPT } ![\text{"MAIN"}] = \text{"Done"}]$
$\land \text{UNCHANGED } \langle response, service\_request, scan\_task\_status, scan\_tasks,$
$key\_to\_be\_served, block\_to\_be\_served, accounts, blocks,$
$stack, inner\_state, inner\_accounts, inner\_blocks\rangle$

$Main \triangleq CreteAccountCall \lor End$

Allow infinite stuttering to prevent deadlock on termination.

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

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

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

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

END TRANSLATION